

Computer Center Users' Guide

507-H

2/27/81

i

1. ORIENTATION.....	5
1.1 ORGANIZATION.....	5
1.1.1 DCRT (Division of Computer Research and Technology).....	5
1.1.2 Computer Center Branch (CCB).....	6
1.1.3 Data Management Branch (DMB).....	6
1.1.4 Computer Systems Laboratory (CSL).....	7
1.1.5 Laboratory of Applied Studies (LAS).....	7
1.1.6 Physical Sciences Laboratory (PSL).....	8
1.1.7 Laboratory of Statistical and Mathematical Methodology (LSM).....	8
1.2 LOCATION.....	9
1.2.1 NIH Campus.....	9
1.2.2 Building 12.....	10
1.2.3 Building 12A.....	12
1.2.4 Building 12B.....	14
1.2.5 Restricted Access Areas.....	16
1.3 TELEPHONE NUMBERS.....	17
1.3.1 Personnel Directory.....	18
1.3.2 Service Directory.....	20
1.4 OPERATING HOURS.....	22
1.5 REGISTERING FOR USE OF SERVICES.....	23
1.5.1 Registered Initials.....	23
1.5.2 Keyword Protection.....	24
1.5.3 Output Boxes.....	25
1.5.4 Account Authorization Forms.....	25
1.5.5 Updating Registration Information.....	31
1.5.6 Terminating Use of Services.....	31
1.6 SERVICE CHARGES.....	34
1.6.1 Batch Charging Formula.....	34
1.6.2 Discount Processing.....	35
1.6.3 TSO (Time Sharing Option).....	35
1.6.4 WYLBUR.....	35
1.6.5 Offline Output Facilities.....	36
1.6.6 Tape and Disk Storage.....	37
1.6.7 Peripheral Services.....	37
1.6.8 Billing Information.....	38
1.6.9 Refund Policy.....	41
1.7 CONTRACTING GUIDELINES.....	43
2. SERVICES FOR USERS.....	47
2.1 CONSULTING.....	48
2.1.1 Tutorial Assistance - Training Unit.....	48
2.1.2 Programming Assistance - PAL Unit.....	48
2.1.2 Using the Programmer Trouble Report (PTR).....	48
2.1.2 Documenting the PTR.....	49
2.1.2 Submitting PTR's from Terminals.....	51
2.1.2 Obtaining Refunds (PTR).....	52
2.1.3 Terminal Problems - Dedicated Equipment Services...	54

HE 20

.3008

:C 73

COLUMBIA UNIVERSITY
THE
LIBRARIES
IN THE CITY OF NEW YORK
DOCUMENTS
SERVICE CENTER

COLUMBIA UNIVERSITY

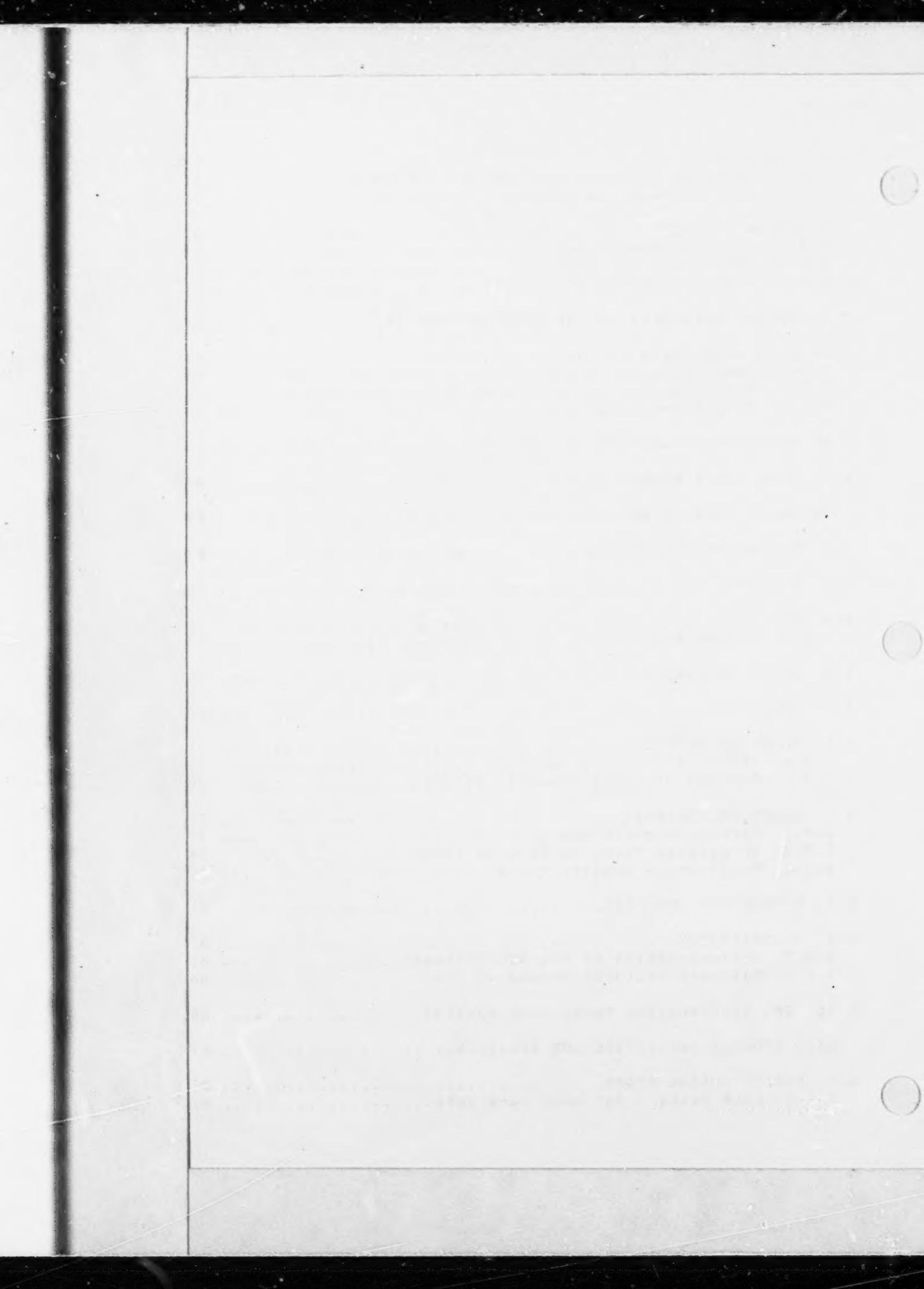
LIBRARIES

APR 20 1981

U. S. DEPOSITORY

COPY

2.2	TRAINING.....	55
2.2.1	Computer Training Courses and Seminars.....	55
2.2.2	Self-Instruction Courses.....	55
2.3	DOCUMENTATION.....	57
2.3.1	Computer Center Publications.....	57
2.3.2	Other DCRT Publications.....	58
2.3.3	Vendor-Supplied Publications.....	59
2.4	USERS' WORK AREA AT THE COMPUTER CENTER.....	60
2.5	DATA CONVERSION.....	61
2.5.1	Data Entry.....	61
2.5.2	Plotting.....	62
2.5.3	Output Handling.....	62
2.6	OUTPUT DISTRIBUTION.....	63
2.7	TEST JOB STREAM.....	64
3.	SOFTWARE FACILITIES.....	67
3.1	OPERATING SYSTEM/370.....	67
3.2	WYLBUR.....	68
3.3	TSO.....	71
3.3.1	TSO/WYLBUR.....	73
3.4	BATCH LANGUAGES.....	74
3.5	UTILITIES.....	76
3.6	PRINTING SERVICES.....	77
3.6.1	Standard Printing Services.....	77
3.6.2	Offline Printing Services (SPOUT).....	78
3.7	GRAPHICS SOFTWARE.....	79
3.7.1	CalComp Subroutines.....	79
3.7.2	Integrated Plotting Package (IPP).....	79
3.7.3	Publication Quality Plots.....	80
3.8	MICROFICHE SERVICES.....	81
3.9	RESTRICTIONS.....	82
3.9.1	Implementation of Non-NIH Software.....	82
3.9.2	Software Features Banned at NIH.....	83
3.10	IMS (Information Management System).....	84
4.	DATA STORAGE FACILITIES AND SERVICES.....	87
4.1	PUBLIC ONLINE DISKS.....	87
4.1.1	FILE Disks - for most data sets.....	87



INTRODUCTION

The Users Guide is published and maintained by the Computer Center for programmers and other technical personnel who use the IBM 370 computing facility. It includes descriptions of each of the services offered and supported by the Computer Center, directions on how to register for their use, and the standards to be followed in using them, as well as directions on who to contact for various kinds of assistance and information.

The Users Guide will be updated so that it reflects the latest picture of computing at the NIH Computer Utility. Between updates to the guide, users will be informed of changes via INTERFACE, the Computer Center's set of technical notes. This version of the Users Guide is updated through INTERFACE Number 92, December 1980.

To get a quick overview of our organization, services, fees, software, and hardware, obtain a copy of the brochure, "The NIH Computer Utility Yesterday and Today." New and prospective users may obtain an orientation package that includes this brochure and other useful items. The brochure and orientation package are available upon request from the Computer Center's Technical Information Office, Building 12A, Room 1017, telephone 301-496-5431.

READER
FICHE

1. ORIENTATION

This section contains introductory material on the organization of the Division of Computer Research and Technology, as well as information on its location, hours of operation, and charges for service. An orientation to other aspects of the Computer Utility, including user services, software, data storage, and hardware, is found in Sections 2 through 5.

1.1 ORGANIZATION

1.1.1 DCRT (Division of Computer Research and Technology)

The Division of Computer Research and Technology provides a variety of data processing services on a fee-for-service basis to the NIH and to other government agencies upon written request.

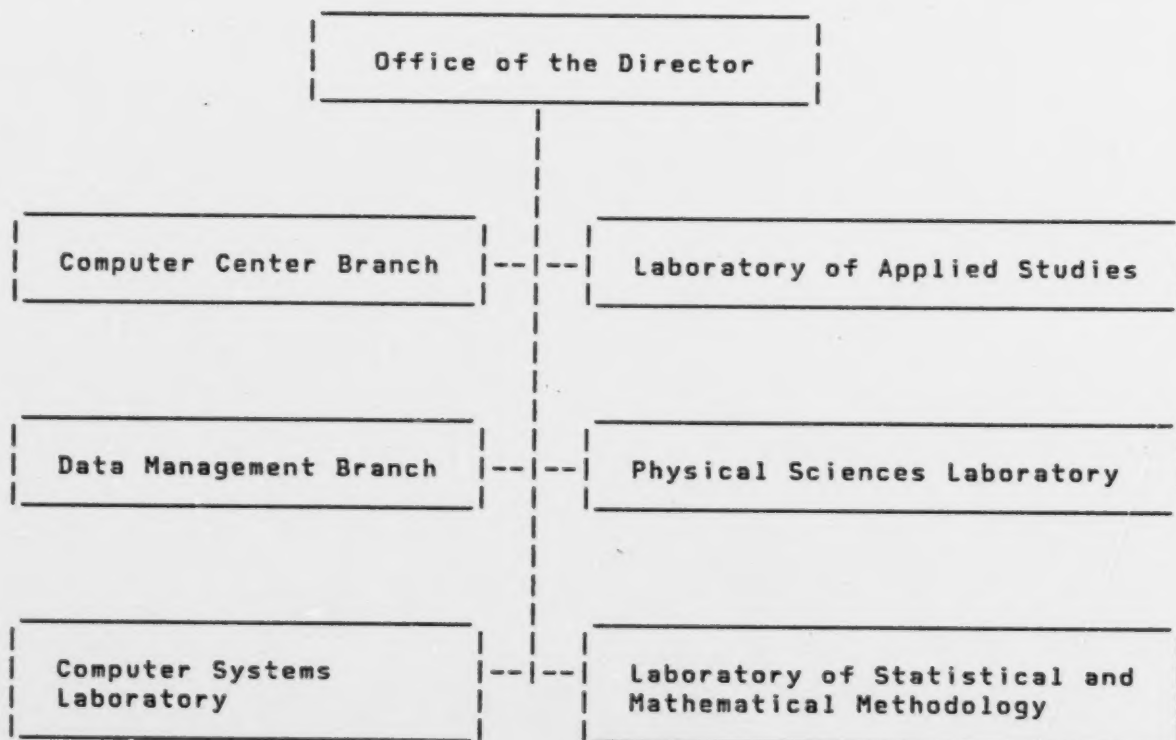


Figure 1.1 DCRT Organization Chart

2/27/81

1.1.2 Computer Center Branch (CCB)

The NIH Computer Center is the branch of DCRT responsible for the operation and performance of the NIH Computer Utility. This involves the design, selection, implementation, maintenance, and operation of a unified computing complex composed of facilities with multiple subsystems and highly sophisticated operating software. It is the mission of the Computer Center to provide the NIH with the most comprehensive and responsive computer services available. The Utility which fulfills this responsibility is financed by charging users for the resources they use.

The NIH Computer Utility encompasses two major facilities, the IBM 370 facility and the DECsystem-10. The two facilities are joined by a data link which can transfer files between them. The MERCURY software makes such transfer easy. Each facility offers a range of online and offline services.

In 1979 the NIH Computer Center became a Federal Data Processing Center under authority of the Brooks Act (PL89-306). Thus it operates as a specialized assistance center to meet the needs of multiple federal agencies. Only services of the IBM System 370 facility are available for use by other agencies.

The services of the Utility's IBM 370 computing facility are described in this document. It is intended to provide users with a complete description of what the facility offers and how to use it effectively. Another publication, the DECsystem-10 Timesharing Guide, provides the same type of information for users of the DECsystem-10. To augment these publications, the Computer Center offers both formal training and informal consultation as well as a complete documentation service to its users.

1.1.3 Data Management Branch (DMB)

The Data Management Branch offers systems analysis and programming support for data management on a fee-for-service basis. The branch provides two types of services: standard software packages for organizations which are willing to conform to standard design approaches, and individual systems analysis and programming service for areas which need them.

The standard packages offered are:

A. MARK IV is a package which enables the development of file management systems at the systems analysis level. It allows experienced programmers to prepare reports, retrieve specific data, perform calculations, and maintain files.

B. IRS (Inquiry and Reporting System) is an advanced, general purpose information system. It gives the user a fast, efficient, easy-to-use technique for extracting information from computer

files; performing basic data processing operations; and producing card, printer, tape, and disk output. It can be learned quickly and used effectively by the programmer and non-programmer alike. The user need only have a knowledge of the data system and some familiarity with data processing techniques. IRS gives the user the ability to respond quickly to short deadlines in an effort to satisfy unscheduled and one-time requests for information as well as recurring report requirements.

C. RMAG (Recursive Macro Actuated Generator), is a program used for the generation of source language coding, data, text, and arbitrary strings. RMAG22, the current version of the powerful macrogenerator, is available in both stand-alone and subroutine callable varieties. RMAG is used to generate source code, commands, or input for all computer languages and systems. The subroutine callable RMAG is available to higher-order and Assembler language programmers, enabling them to add this powerful macrogenerator to their own systems by means of simple subroutine calls. Although the current implementation runs on an IBM 370 under OS or VS, RMAG may be used to generate programs for most other computers.

D. SLANG is a powerful compiler used to generate block-structured 370 Assembler language source code. It is used by adherents of structured programming as well as by those who want to understand all parts (i.e., instructions, data, tables, etc.) of the programs they write. Using SLANG, it has been possible to write super-efficient, complex systems swiftly.

E. Autotab II is an easy-to-use software package designed to create and update tables of numbers. Even non-programmers can use the system, which requires only basic information (about rows, titles, column headings, initial values, and calculations needed) to produce attractive final reports. Related tables can be consolidated so that changes in one table can be "rippled" through all others in the group.

1.1.4 Computer Systems Laboratory (CSL)

The Computer Systems Laboratory provides expertise in laboratory automation for the NIH. The CSL provides consultation and collaboration in the design and implementation of specialized computer systems for laboratory automation, patient monitoring, and other computer applications where use of the facilities and services of the central computer utility is not appropriate.

1.1.5 Laboratory of Applied Studies (LAS)

The Laboratory of Applied Studies applies the disciplines of biostatistics, medicine, applied mathematics, computer science, and biomedical engineering to current problems in health research. Major projects have concerned the development of mathematical models and computing systems related to noninvasive

diagnostic and patient monitoring techniques.

1.1.6 Physical Sciences Laboratory (PSL)

The Physical Sciences Laboratory studies biological phenomena in terms of theoretical physics and chemistry and offers assistance to others interested in such problems.

1.1.7 Laboratory of Statistical and Mathematical Methodology (LSM)

The Laboratory of Statistical and Mathematical Methodology uses the methods of mathematical statistics, mathematics, and computer science as tools for biomedical research and provides consultative and collaborative services to NIH scientists and researchers. The laboratory maintains and develops software and methods for biomedical modeling and data analysis and conducts research in new areas of biomathematics, biostatistics, biomedical modeling, and computation.

In addition to computer programs developed at NIH, LSM's program library contains the following packages:

- A. SAS - Statistical Analysis System
- B. SPSS - Statistical Package for the Social Sciences
- C. BMD - UCLA Biomedical Computer Programs
- D. BMDP - UCLA Biomedical Computer Programs (P series)
- E. PSTAT - Princeton Statistical Package
- F. IMSL - International Mathematical and Statistical Library

MLAB, an online modeling system, and C-LAB, a cluster analysis system, are two other facilities. MLAB and C-LAB run on the DCRT DECsystem-10 timesharing system. Another program, MODELAIDE, is available for modeling under TSO on the System/370.

The staff of LSM is prepared to answer questions concerning the use of these packages. The staff is also available for consulting in the areas of statistics, mathematics, and computer science.

1.3 TELEPHONE NUMBERS

The charts in this section serve as a directory to people and services. Telephone numbers and personal names are listed in the "Personnel Directory" and "Service Directory" only. Elsewhere in this guide, reference is made only to office titles.

The area code is 301.

For users outside the local calling area, FTS service (if available) may be used to call offices and terminal systems.

1.3.1 Personnel DirectoryA. COMPUTER CENTER OFFICES

Office	Person	Bldg.-Room	Telephone
Chief, Computer Center	Joseph Naughton	12B	496-5381
Head, Systems Research and Development Sect.	Robert Brunelle	12B	496-1708
Head, User Services and Assistance Sect.	Fran Halverson	12A - 1017	496-5525
Programmer Assistance and Liaison Unit	Jim Gangler	12A - 1017	496-5525
	Sheryl Katz	12A - 1017	496-5525
	Pete Morton	12A - 1017	496-5525
	Kathy Scalzi	12A - 1017	496-5525
	Tom Wagner	12A - 1017	496-5525
	Harriet Weltman	12A - 1017	496-5525
	Kay Wiggins	12A - 1017	496-5525
	Keith Younger	12A - 1017	496-5525
Training Unit	Pat Logan	12A - 1031	496-2339
	Linda Alger	12A - 1031	496-2339
	Libby D'Antuono	12A - 1031	496-2339
	Thelma Smith	12A - 1031	496-2339
Technical Information Office	Patricia Cleveland	12A - 1017	496-5431
	Ruth Boxall	12A - 1017	496-5431
	Dorothy Smith	12A - 1017	496-5431
	Leslie Barden	12A - 1013	496-5431
Dedicated Equipment Services	Gene Gruden	12A - 1033	496-2248
Computer Operations	Wes Farmer	12A - 1001A	496-5815
Output Distribution	Robert Rizzo	12A - 1000	496-6183
Output Handling	William Owens	12A - B61	496-4452
Data Conversion Services Control Point	Irene Davis	13 - 2E50	496-4451
	Vicky Williams	13 - 2E50	496-4451
Info. Media Library	Grace Rizzo	12 - 1100	496-6021
Plotting Services	Ann Pole	12A - B61	496-5430
DECsystem-10 Services	Glenn Ricart	12B	496-4823

1.5 REGISTERING FOR USE OF SERVICES

Anyone who wishes to use Computer Center services must obtain a DCRT account number using the appropriate form described in Section 1.5.4 "Account Authorization Forms." These forms are also used to request user, project, and storage initials and DECsystem-10 project programmer numbers. In addition, they are used to register for an Output Distribution Services output box.

1.5.1 Registered Initials

Proper use of accounts and initials provides a form of protection. Each user should have his own set of user initials and should not allow them to be used by anyone else. This policy will be enforced.

A. User Initials

In addition to the account number, each user will be registered for a set of user initials. One user may be authorized to use one or more account numbers, and an account may have one or many authorized users. All authorization of accounts and initials is done by the Project Control Office.

Each registered user is permitted to receive one set of computer documentation. Non-users can be given only introductory orientation information and cannot register for most computer training courses.

B. Project Initials

Project initials permit a group of users to work together on a project running jobs and using the same data sets without divulging their individual keywords. Project initials may be used to run batch jobs, and to access and store data sets. They cannot be used to sign on to WYLBUR or TSO.

C. Storage Initials

Storage initials can be used to store data sets and to rent tapes and disks. They cannot be used to sign on to interactive systems or run batch jobs.

Storage initials should be used to save data which will be shared by a group of users since the keywords belonging to the individuals (which guarantee account/initials integrity) will not have to be divulged.

If tapes are assigned to storage initials, a keyword must be set for the initials.

1.5.2 Keyword Protection

Keywords protect each account/initials set from unauthorized use. A keyword is made up of three text characters other than the user's initials (blanks are not permitted). If a keyword contains a character which cannot be entered from a certain terminal, that terminal cannot be used until the keyword is changed.

Users are required to set a keyword for each account and initials combination they use. The keyword protects the account and initials against unauthorized use on WYLBUR and TSO. WYLBUR data sets cannot be SCRATCHed from a terminal by anyone who does not know the keyword, and specially designated private data sets cannot even be USED. This facility protects TSO users against unauthorized access and destruction of their data sets. Please note that the keyword is associated with the account and initials, not the data sets. This means that the user must remember only his current keyword, not the one in effect when a data set was created.

The keyword also protects the user's account and initials against unauthorized use by batch jobs. A correct /*KEYWORD card must be included in every job that is run with the user's account and initials. WYLBUR and TSO prompt for keyword information from the user and supply it directly to the system, so the /*KEYWORD card need not be included if the job is submitted through either of these systems. See Section 6.10.1 for further information on /*KEYWORD cards.

Forgotten keywords will be divulged only by formal written request by the user or to individuals who can absolutely identify themselves. The requests should be addressed to: Project Control Office, DCRT, and must specify the account number and the registered initials which are protected by the forgotten keyword. The request must be signed personally by either the registered sponsor of the account or the individual whose registered initials the keyword protects. After verifying the signature against the one that is in the official Project Control Office files, the requested keyword will be divulged to the sponsor by telephone using the number on record in the project file. When the request to divulge a forgotten keyword is signed by an individual other than the account sponsor and is hand carried to the Project Control Office, the requested keyword will be divulged to the requestor only if the identity of the individual can be absolutely verified (i.e., employee identification card, driver's license, etc.). Telephone requests for forgotten keywords will not be honored under any circumstances. THERE WILL BE NO EXCEPTIONS TO THIS POLICY.

Keywords should be changed frequently. They can easily be set and changed through WYLBUR or TSO or by using the KEYSET program described in Section 12.18. For further information, see the WYLBUR Fundamentals or General Editing manual or the NIH TSO Reference Manual.

RETURN TO: Project Control Office, DCRT Building 12A, Room 3009		ADDRESS/INFORMATION CHANGE NOTICE	
A. REQUESTER (Contact)			
NAME	ADDRESS	TELEPHONE	
B. REASON FOR CHANGE			
1. ☐ Change of address and/or phone as indicated below in Section C, "NEW INFORMATION"			
2. ☐ Invalidate (complete applicable items):			
a	b	c	d
ACCOUNT	INITIALS	REGISTERED NAME (IN FULL)	DELETE AS REG. USER
			☐
			☐
			☐
3. ☐ Cancel:			
a. ☐ All publications			
b. ☐ All publications except INTERFACE			
4. ☐ Reassign as project initials (fill in below all information on assignee and forward to him for signature)			
INITIALS	TO (Name)		
ACCOUNT	ADDRESS	TELEPHONE	
SIGNATURE OF ASSIGNEE			
5. ☐ Other (Specify):			
C. NEW INFORMATION (to be filled in if 1 or 3b above is checked)			
NAME	BUREAU/INSTITUTE/DIVISION	TELEPHONE	BOX NO.
ADDRESS		CONTRACTOR (if on NIH contract)	
D. AUTHORIZATION			
DATE	SIGNATURE	EFFECTIVE DATE	
NOTES: This form cannot be used to initiate new accounts or initials. For this purpose, obtain a "DCRT Account Authorization" form (NIH-1767-1) from the Project Control Officer, Building 12A, Room 3013, Telephone 496-6146.			
NIH-1767-2 Rev. 9-74			

Figure 1.11 Address/Information Change Notice

1.6 SERVICE CHARGES

A schedule of rates governing the various kinds of services offered by DCRT has been established under the NIH Service and Supply Fund (Revolving Fund). For billing purposes, each customer-defined project is identified by a DCRT account number. This account number must be used on all requests for services, as well as on every JOB statement and at the beginning of every terminal session. In addition, everyone who personally uses any computer services must obtain a set of registered user initials. Users of data processing facilities are billed monthly. For conditions under which refunds will be made, see Section 1.6.9.

1.6.1 Batch Charging Formula

The Computer Center charges for the use of its central facilities at a rate of \$.50 per machine unit with a minimum charge of \$.25. This charge is computed in the following way:

$$MU = (1 + .371D + .0196T + .0012R)(N * C + .00017I) + .00174U$$

where

MU = machine units
D = number of mountable disk drives
T = number of tape drives
R = region size in K units
N = .3 for the 370/168-3
 .6 for the 3033
C = CPU time
I = I/O count
U = unit record transfers

The I/O count is approximately the number of blocks read and written from tape and disk data sets.

The number of unit record transfers is the sum of the number of cards read, the number of cards punched, and the number of lines spooled or printed. (This is \$.87/1000 for such transfers). The region size used is charged for rather than the region size requested.

The CPU time requirement for a particular job varies drastically between installations. These variations depend upon the types of hardware and level of the operating system being used as well as internal modifications made by the installations.

1.6.9 Refund Policy

If a job aborts due to operator error, hardware failure, or software failure (systems software, not the user's program), the user may request a rebate for the run that aborted. The "Programmer Trouble Report and Request for Refund," Form NIH-1654 is used for this purpose. (See Section 2.1.2 for directions). Users unfamiliar with the standards of the Computer Center should consult Section 6.1, "Standards for Batch Jobs."

The following is a summary of the regulations governing when refunds will be given. A user would be wise to keep these conditions in mind when running a job as well as requesting a refund.

1. All evidence necessary for investigation of the problem (such as source, tapes and/or disks, and data sets), must be preserved unchanged.
2. No refund will be given for less than \$5.00.
3. If a terminal session or batch job is ended because the system crashes, no charge is made and, therefore, no refund is necessary.
4. No refund will be granted for a job which violates installation standards, even if it fails because of operator error, hardware failure, or a failure of system software.
5. All Class C jobs should be step-restartable. If a Class C job aborts and has no step restarts, refund will be made only for the step that aborted. For a very large step, the maximum refund will be for one hour of processing.
6. Users are responsible for determining reasonable CPU time requests for their jobs. A Class C job which runs for the maximum time limit may cost thousands of dollars. No refund will be granted for a program looping situation which results in a 322 ABEND.
7. If more than 3000 cards are read in at a remote batch workstation, no refund will be given in case of an I/O error during the input process.
8. Unless a preassigned tape was used, the PAL Unit must be notified of any refund request for SPOUT processing before the tape is recycled. This is necessary since scratch tapes mounted for SPOUT output are kept for no more than 24 hours after processing. The user may submit the PTR through WYLBUR's PTR command to insure prompt delivery or call the PAL Unit to notify them of the problem.
9. The user must insure that a job runs successfully before other jobs that are dependent on it are allowed to run.

(DSSUBMIT can be used to accomplish this; see Section 12.11 "Program to Submit Batch Jobs.") No refund will be allowed for any job that fails to run because it was dependent on another job which did not run successfully. This applies even if the original job failed because of a system or operator error.

10. No refund will be allowed for entire jobs which have interdependent steps with no condition code testing. The maximum refund allowed in case of a system or operator error is for the step that terminated abnormally.
11. Refunds will be given only for the original job and one rerun. Do not run a job repeatedly hoping that the problem will disappear. If a refund is requested because the job failed but the rerun (with no changes) was successful, the documentation from both runs must be submitted to justify the refund.
12. No refund will be given for a job which specifies a message level that causes messages to be suppressed, e.g. MSGLEVEL=(0,0).
13. No refund will be given for any job which fails because of I/O errors encountered when using a "special" tape (one not owned by the Computer Center).
14. The Computer Center may choose to rerun a job at its own expense after correcting or bypassing the problem which caused the original job to fail. When this is done, no refund will be given.
15. No refund will be granted for a looping TSO session with WARN OFF specified.
16. Programs which are debugged and running on a production basis should include a //SYSUDUMP or //SYSABEND statement because many errors which occur in this type of job can be diagnosed only with a dump. If the job runs properly, the dump statement costs nothing, but if it fails, a dump may be needed to find the problem and give a refund.

2. SERVICES FOR USERS

Helping users is one of the most important functions the Computer Center performs. This function is carried out by the User Services and Assistance Section, which performs a wide variety of informational and programmer assistance services. The information supplied by INTERFACE and the documentation supplied through the Technical Information Office keep users informed and up-to-date. Programmer assistance services, such as the PAL Unit, Training Unit, and Technical Information Office, help the user in his daily work by supplying documentation, training, help with programming problems, etc. All users of the central computing facility and its remote computing facilities should become familiar with these services.

Dedicated Equipment Services assists users in selecting and maintaining interactive terminals and RJE workstations.

2.1 CONSULTING

2.1.1 Tutorial Assistance - Training Unit

The Training Unit advises users who need tutorial help, both on a walk-in basis and over the telephone. Staff members will help users in any area in which they are competent. Currently, aid can be given in WYLBUR, FORTRAN, PL/I, and COBOL. They can also provide general information about the basic use of computers, hardware, and software at the Computer Center.

2.1.2 Programming Assistance - PAL Unit

The PAL (Programmer Assistance and Liaison) Unit provides walk-in and telephone consultation to the users of the Computer Utility. If a problem cannot be resolved on the telephone or in a reasonably short time during a walk-in visit, or if a refund is in order, a PTR (Programmer Trouble Report) must be submitted, either on a PTR form or through WYLBUR's PTR command.

Before contacting the PAL Unit for help with a problem, please obtain and consult the applicable current documentation available from the Technical Information Office.

IMPORTANT NOTE: If you think that an action you are taking at a terminal or a batch job you are running is causing the system to crash, contact the PAL Unit immediately.

2.1.2 Using the Programmer Trouble Report (PTR)

The "Programmer Trouble Report and Request for Refund", or PTR, (Figure 2.1) is a form used to communicate with the Computer Center. User problems and refund requests which are submitted to the PAL Unit must be accompanied by a PTR. It may also be used to communicate suggestions and comments informing the Center of users' needs. The information from this form helps the staff to formulate future policies, plan systems changes, and inform users of common trouble areas.

For greater convenience and when immediate attention is needed, PTR's can be submitted from any interactive terminal by using WYLBUR's PTR command. The information from a PTR submitted from a terminal is listed automatically at the Computer Center on a 5-part PTR form.

PTR forms may be obtained in person or by telephone request from the Technical Information Office. Users at remote locations may keep a supply to mail in. The original 5-part PTR form is the only type that can be accepted; reproduced copies will be returned to the sender.

When a PTR is received by the PAL Unit, it is examined to insure that all pertinent documentation is included. It is

directed to the responsible person within the PAL Unit or forwarded to the group responsible for the area in which the problem occurred. The Systems Team, Operations Unit, and IBM representatives receive most of the PTRs which cannot be handled within the PAL Unit. If the final answer to a PTR will be delayed because only a long-range solution is possible or because it must be forwarded to another group for handling, the user will be contacted and informed of this fact.

Whenever possible, the PAL Unit will provide a temporary solution or bypass for the problem. For non-critical problems and questions, one or two weeks may be required before a response is given. If the problem is critical, please indicate this on the PTR, and if at all possible, a bypass will be given within two working days. Please do not indicate that a problem is critical unless it is severely impacting your work and cannot wait. The two-day response to critical problems can be offered by the PAL Unit only if it is not abused. Permanent fixes typically take longer since they must be supplied by the software developer or vendor.

Once the problem is permanently corrected, the second (green) copy explaining the solution is returned to the user's computer output box. If a refund has been requested, the fourth (pink) copy is returned after the refund request has been processed. The third (yellow) copy is retained by the Project Control Office which processes refunds, and the last (gold) copy is kept by the IBM representatives.

2.1.2 Documenting the PTR

The description of the suggestion, problem, or complaint should clearly, but briefly, explain the area of concern. Pertinent computer listings (hard copy terminal output, printed job output, dumps, etc.) should be sent to the PAL Unit. All necessary documentation and data for the problem investigation (e.g. source, data sets, disks and/or tapes), must be frozen until the PAL Unit has studied the problem; this is critical in cases where the problem must be reproduced to obtain the resolution.

Users should be certain to include their names on PTR's since the PAL Unit frequently must contact the individual before being able to resolve a problem.

The PAL Unit will not attempt to help users with any jobs which suppress messages: e.g. MSGLEVEL=(0,1).

The amount of documentation needed depends on the severity and complexity of the problem. For example, if a job fails and a rerun with no changes is successful, the failing and successful jobs are all that is needed for documentation. On the other hand, if there seems to be a problem with the way COBOL handles a particular verb, the supporting documentation would include:

- 1) source for the program (an online data set or card deck)
- 2) an Assembler language listing of the object program (PMAP and DMAP)
- 3) a link-edit map
- 4) the failing run (with full dump if there was a system ABEND)
- 5) a card deck or data set with the JCL needed to recreate the problem

A full dump will be needed to document any job which terminates with a system ABEND. For information on how to obtain dumps, see Section 6.5.15 "SYSUDUMP and SYSABEND Statements." Users with questions about the documentation needed for particular problems should call the PAL Unit for advice.

SPOUT

Any problem involving an unassigned SPOUT tape must be reported immediately since scratch tapes mounted for SPOUT output are kept no more than 24 hours. The PTR may be submitted through WYLBUR to insure that it is delivered promptly.

Terminal Sessions

The hardcopy listing from the terminal session must be included when submitting a PTR. The entire listing, including LOGON and LOGOFF, must be included, especially if the user is requesting a refund.

CRT Terminal Sessions

Users experiencing problems with CRT terminal sessions must be particularly careful to include in their problem description specific instructions describing how the problem may be re-created. A TSO CLIST, a WYLBUR SCRIPT file, hardcopy documentation from a printer attachment or a re-creation at a hardcopy terminal may be used to supplement this information.

2.1.2 Submitting PTR's from Terminals

A PTR may be submitted remotely through WYLBUR (and WYLBUR under TSO):

1. Place all documentation pertinent to the problem in OUTPUT HOLD. A PTR cannot be processed by the PAL Unit unless all documentation is available.
2. In response to any ? prompt from WYLBUR, enter the command, PTR.
3. WYLBUR will offer guidelines and prompt for all required input. For more detailed information concerning the appropriate response, a question mark (?) may be entered after any prompt.

When WYLBUR prompts for the problem description, the answer should be concise and detailed. The problem description should include:

- Whether the job incurring the problem can be rerun by the PAL Unit
 - Names and volume serial numbers of all data sets related to the problem
 - Any information which may help to resolve the problem.
4. WYLBUR will ask whether the problem is SEVERE or CRITICAL to determine the priority to be given to the response.
 5. Before the PTR is submitted, the text will be listed for review and editing. Then there is a final chance to cancel the PTR before it is submitted.

2.1.2 Obtaining Refunds (PTR)

To obtain a refund, a user must submit a PTR to the PAL Unit, either via WYLBUR's PTR command or on a "Programmer Trouble Report and Request for Refund," Form NIH-1654 (Figure 2.1). All supporting documentation (source listings, decks, dumps, entire terminal listings, etc.), should be sent to the PAL Unit. Requests for refunds must be received by the PAL Unit within 3 working days after the run has been completed. The documentation needed is described above under "Documenting the PTR".

See Section 1.6.9 for a summary of the conditions under which a refund will be granted. A refund request which does not meet the criteria listed will be denied.

PROGRAMMER TROUBLE REPORT AND REQUEST FOR REFUND SYSTEM/370	 Return to: Programmer Assistance & Liaison Unit Computer Center Branch, DCRT National Institutes of Health Building 12A, Room 1017 Bethesda, Maryland 20205			
1. IDENTIFICATION (All information must be provided)				
NAME/MAILING ADDRESS	ACCOUNT NO./INITIALS	BOX NO.	PTR NO.	
	TELEPHONE NO.	DATE SUBMITTED		
ORGANIZATION-CONTRACT CO.				
2. TYPE OF REPORT (Refund requests must be submitted within 3 working days of failure. Please complete PART 1 above and supply appropriate documentation.)				
☐ Problem ☐ Complaint ☐ Suggestion		☐ Refund Request		
Please insure that appropriate documentation/information is attached or included in the problem description below. Consider the following: terminal sheets, batch job output, source listing, data set names and locations, dump, contains private data, has job rerun successfully, permission to rerun.		Date of run _____		
		Final time _____		
		Machine Units _____		
		Amount \$ _____		
PROBLEM DESCRIPTION AND RESOLUTION				
SAMPLE ONLY				
COMPUTER CENTER ENTRIES (DO NOT WRITE BELOW THIS LINE)				
OPERATIONS	ACCOUNTING			PROJECT CONTROL
SAD _____	CAUSE OF ERROR	TYPE OF CHARGE	REFUND CONCLUSION	Refund will appear on invoice dated: _____
_____	☐ 1 Hardware ☐ 2 Software ☐ 3 Operations ☐ 4 Accounting Transfer ☐ 5 Miscellaneous ☐ User Error	☐ 21 Batch ☐ 22 Terminal (WYLBURT SO) ☐ 23 MICRO ☐ 32 DECsystem-10 ☐ 41 On-Line Storage ☐ 42 Dedicated Equipment ☐ Other (Specify)	☐ Approved ☐ Denied	
SVC (DYNA) _____			MACHINE UNITS	Entered by: _____
_____			AMOUNT	Date: _____
_____			DATE	
_____			CCB AUTHORITY	
NIH-1654-1 REV 12-80				

Figure 2.1 Programmer Trouble Report and Request for Refund (NIH-1654)

2.1.3 Terminal Problems - Dedicated Equipment Services

Dedicated Equipment Services can supply information to assist users in deciding what model of interactive terminal or RJE workstation to order. When telephone line problems occur or when a terminal or workstation appears to be malfunctioning, they can advise the user on what to do. They handle all arrangements for repairs to terminals and workstations ordered through the NIH Computer Center.

2.3 DOCUMENTATION

The Technical Information Office is responsible for the distribution of Computer Center publications, IBM manuals, DECsystem-10 documentation, and some DCRT publications. One set of documentation is available to each user of Computer Center services who has an account number and personal set of registered initials. People who are not registered users may obtain copies of INTERFACE, the "Computer Training Courses and Seminars" brochure, "The NIH Computer Utility Yesterday & Today," and "Introducing the NIH Computer Utility."

Users may obtain manuals immediately by going to the Technical Information Office, or manuals may be requested by telephone or mail and sent to the user. It is possible to call the Technical Information Office to check if a specific document or its latest version is in stock. There is no charge for any of these publications.

Through the Automatic Documentation Service, a profile of each user's documentation requirements is kept by the Technical Information Office, and users automatically receive updates to their sets of Computer Center and vendor-supplied publications. Each year the user must return an "Automatic Documentation Renewal" form, which is published in the December issue of INTERFACE, to continue receiving publications.

2.3.1 Computer Center Publications

The following Computer Center publications are available from the Technical Information Office. Updates will automatically be sent when issued.

1. INTERFACE - The Computer Center's series of technical notes. Every user should receive INTERFACE and keep it available for reference.
2. Computer Training Courses and Seminars
3. Computer Center Users Guide
4. WYLBUR Fundamentals
5. WYLBUR Fundamentals Reference Handbook
6. WYLBUR General Editing
7. WYLBUR Document Formatting
8. WYLBUR Batch Processing
9. WYLBUR Command Procedures
10. WYLBUR Reference Handbook

11. WYLBUR Master Index
12. NIH ISO Terminal User's Guide
13. NIH ISO Reference Manual
14. A User's Guide to the Integrated Plotting Package
15. Edit Format Utility Package

Single copies of the following Computer Center publications are available:

1. The NIH Computer Utility Yesterday & Today
2. Introducing the NIH Computer Utility
3. Terminal Operator's Guide
4. NIH Job Control Language (JCL) Primer
5. Remote Job Entry Workstation Guide
6. EZFORM/PTFORM User's Guide
7. PALTAPE--A Utility for Handling Tape Problems
8. RHB Routines
9. Assembler G User's Guide
10. ALP: Assembler Language Preprocessor User's Guide
11. PROGLOOK User's Guide
12. Programming for the CalComp Plotter on the IBM 370 Facility at NIH
13. Easy Reader--A Private Procedure Library Facility
14. User's Guide for IMS/VS at the NIH Computer Center

2.3.2 Other DCRT Publications

The following publications supported by other laboratories and branches of DCRT are distributed by the Computer Center's Technical Information Office:

1. DCRT Mathematical and Statistical Program Manual - describes the programs in the DCRT Math-Stat Program Library. (Section 1.1.7 lists the packages supported by the Laboratory of Statistical and Mathematical Methodology.)

3. SOFTWARE FACILITIES.....	67
3.1 OPERATING SYSTEM/370.....	67
3.2 WYLBUR.....	68
3.3 TSO.....	71
3.3.1 TSO/WYLBUR.....	73
3.4 BATCH LANGUAGES.....	74
3.5 UTILITIES.....	76
3.6 PRINTING SERVICES.....	77
3.6.1 Standard Printing Services.....	77
3.6.2 Offline Printing Services (SPOUT).....	78
3.7 GRAPHICS SOFTWARE.....	79
3.7.1 CalComp Subroutines.....	79
3.7.2 Integrated Plotting Package (IPP).....	79
3.7.3 Publication Quality Plots.....	80
3.8 MICROFICHE SERVICES.....	81
3.9 RESTRICTIONS.....	82
3.9.1 Implementation of Non-NIH Software.....	82
3.9.2 Software Features Banned at NIH.....	83
3.10 IMS (Information Management System).....	84

3. SOFTWARE FACILITIES

3.1 OPERATING SYSTEM/370

The software servicing the IBM 370 system is composed of the IBM MVS (Multiple Virtual Storage) operating system and a highly customized version of JES2 NJE (Job Entry Subsystem 2 Network Job Entry) which incorporates the NIH-developed Shared Spool capability. JES2 NJE allows selected jobs, operator commands, messages, SYSOUT data sets, and accounting information to be transmitted between communicating MVS operating systems that are connected by telecommunication lines. These MVS systems may be located at the same site or at geographically separate sites.

MVS vastly expands the apparent amount of storage which is available to each task and uses the computer's memory more efficiently than non-virtual systems. This is possible since only the active portions of programs are kept in core. JES2 acts as the batch job scheduler and controller for the MVS operating system.

This complex operating system presents a unified system interface to the user. Shared files and work queues make it possible to balance the workloads of the various subsystems and to permit service to resume promptly even if one machine is out of service for an extended period of time. This is done by switching the services usually provided by the failing machine to the others.

Users interested in obtaining an understanding of virtual storage should obtain a copy of the IBM student text Introduction to Virtual Storage in System 370, Form No. GR20-4260, from the Technical Information Office.

3.2 WYLBUR

WYLBUR is a computer program that functions as an online interactive text editor. Terminals connected by telephone lines to NIH's IBM System/370 computer system allow the user to communicate with WYLBUR. (For information on interactive terminals, see Section 5.1.) The user uses a collection of predefined instructions or commands to employ the facilities available in WYLBUR. The editing features of WYLBUR provide an easy-to-use mechanism for creating, changing, searching and displaying all kinds of text, such as computer programs and data, letters, proposals, reports, manuals, and lists. Between interactive terminal sessions, data sets can be stored on online FILE, PDS, or TMP disks (see Section 4.1). WYLBUR's batch processing facilities allow the user to submit a computer program to the batch job stream and examine the output at the terminal. The document formatting facility has capabilities such as making the lengths of text lines even; automatically dividing the text into pages; and generating page numbers, headings, footings, tables of contents, and indices. WYLBUR's command procedure facility provides commands to evaluate expressions, execute collections of stored commands, and perform comparisons and branching.

The current version of WYLBUR, known as NIH Extended WYLBUR, was completely developed, designed and implemented at the National Institutes of Health in Bethesda, Maryland. It is a successor to the original WYLBUR, which was developed at the Campus Facility of the Stanford University Computation Center and modified and extended by the NIH Computer Center to meet the needs of the NIH user community.

An individual interested in using WYLBUR must have a valid account number and set of initials, registered with the Project Control Office.

Publications

The following publications about WYLBUR are available through the Technical Information Office:

A. Fundamentals

This manual gives an overview of the WYLBUR system by defining a basic set of commands in each of four areas: editing commands, document formatting, batch job processing, and command procedures. The information given is sufficient for most users to begin to use WYLBUR effectively.

B. Fundamentals Reference Handbook

This manual gives the syntax (form) of the commands described in the Fundamentals manual, but does not contain descriptions of the commands.

C. General Editing

This manual defines all the editing commands available in WYLBUR. Users who need more advanced editing facilities than those described in the Fundamentals manual will find this manual useful.

D. Document Formatting

This manual defines the commands for making the lengths of lines of text even and describes the more advanced facilities that use marker commands to paginate documents and generate headings, footings, tables of contents and indices. It also describes how to print using various fonts, mathematical symbols, Greek letters, and superscripts and subscripts.

E. Batch Processing

This manual describes the more advanced tools for submitting batch jobs and retrieving output.

F. Command Procedures

This manual describes how to execute a collection of prestored commands and statements, and the provisions for calculating, testing, and controlling the flow of execution. It assumes the user has a basic knowledge of some programming language and is familiar with other aspects of WYLBUR.

G. Reference Handbook

This manual contains the syntax (form) of all WYLBUR commands, but does not contain descriptions of the commands.

H. Master Index

This manual provides an index for the following manuals which describe the WYLBUR system: Fundamentals, General Editing, Document Formatting, Batch Processing and Command Procedures.

I. Edit Format Utility Package

This manual describes a package of programs for reading and writing edit format data sets from TSO programs and batch jobs. WYLBUR creates data sets in edit format.

J. Terminal Operator's Guide

This manual describes and illustrates each of the interactive terminals made available through the Computer Center. These include the CleanType 45, T1222, and NIH7000.

K. INTERFACE

The series of technical notes for users, published by the Computer Center.

3.3 TSO

The IBM OS/MVS Time Sharing Option (TSO) is a general purpose timesharing system which permits interactive programming from keyboard terminals in FORTRAN, COBOL, PL/I, and Assembler language. A programmer may direct I/O to the terminal and thus control the course of the processing while the program is executing. TSO also provides a Remote Job Entry facility for submitting batch jobs.

TSO enables the user to manipulate files from a terminal. All data files to be accessed must be stored on the online disks. Data sets created under TSO are placed on the TMP disks unless this default is overridden. (All access methods valid for the language being used are supported under TSO.) Before a data set saved from WYLBUR or the batch can be accessed through TSO, it must be entered in the TSO catalog. TSO cannot access EDIT format data sets.

TSO should be used for applications that allow the user to interact reasonably with the program. A program is inappropriate for TSO if it asks a user to enter some data and then begins extensive file searches involving large amounts of I/O or begins lengthy computations consuming large amounts of CPU time. Such applications are more expensive in TSO than in the batch and may degrade system response for all TSO users.

An online HELP data set provides users with up-to-date information about any TSO command, subcommand, or operand.

Publications

The following publications about TSO are available through the Technical Information Office:

A. NIH TSO Terminal User's Guide

This manual should be read by all programmers who are going to use TSO for the first time. It gives an overview of the major TSO operations and explains how to conduct a session at the terminal. It is a preface to the IBM manual, OS/VS2 TSO Terminal User's Guide, Form No. GC28-0645.

B. NIH TSO Reference Manual

This manual, which incorporates the IBM manual, OS/VS2 TSO Command Language Reference Manual, Form No. GC28-0646, describes how to use the commands, subcommands, and operands of the TSO language. The TSO HELP data set contains this same information for immediate online access.

C. IBM OS/VS2 TSO Guide to Writing a Terminal Monitor Program or a Command Processor, Form No. GC28-0648

This publication contains information needed by a programmer writing a terminal monitor program or a command processor for TSO.

D. Terminal Operator's Guide

This manual describes and illustrates each of the interactive terminals made available through the Computer Center. These include the CleanType 45, T1222, and NIH7000.

E. INTERFACE

The series of technical notes for users, published by the Computer Center.

3.3.1 TSO/WYLBUR

WYLBUR is available as a command processor under TSO. Users can edit data sets with WYLBUR and process them with TSO programs without leaving TSO/WYLBUR. Also TSO CLISTs can contain WYLBUR commands.

The WYLBUR Fundamentals and General Editing manuals describe the capabilities of WYLBUR under TSO. For a list of documents describing WYLBUR, see Section 3.2.

3.4 BATCH LANGUAGES

The NIH Computer Center supports several batch languages, offering through them a wide range of programming capabilities. Details on how to use batch languages at the NIH Computer Utility are given in Sections 7-10.

Assembler Language

The IBM Operating System Assembler is a symbolic programming language used to write programs for the IBM System/370. The language provides a convenient means for representing the machine instructions and related data necessary to program the IBM System/370. Assembler language is generally used for system programming where machine-dependent operations are required that cannot be performed using one of the higher-level languages such as COBOL or FORTRAN. The Assembler program processes the language, provides auxiliary functions useful in the preparation and documentation of a program, and includes facilities for processing the Assembler macro language.

COBOL

COBOL (COmmon Business Oriented Language) is a programming language similar to English which is designed for data processing oriented programming applications. COBOL is used primarily for data manipulation, I/O, report generation, and basic mathematical functions. COBOL was developed by the Conference on Data Systems Languages. The COBOL compiler used at the Computer Center supports the 1974 and 1968 American National Standards (X3.23-1974 and X3.23-1968 respectively), as approved by the American National Standards Institute.

FORTRAN IV

FORTRAN (FORmula TRANslation) is one of the most commonly used programming languages. It is especially useful in writing programs for scientific, engineering, and biomedical applications that involve mathematical computations. When augmented by a subroutine package such as the RHB routines, it is capable of handling scientific data processing applications. FORTRAN is in such widespread use that the American National Standards Institute has defined a standard FORTRAN language. Most implementations of FORTRAN include all the features of the standard language plus some additional capabilities.

The users of FORTRAN have available to them a large program library. This library includes the RHB data processing routines and many others. A description of the RHB routines can be obtained from the Technical Information Office.

PL/I

PL/I (Programming Language I) is a multi-purpose programming language for use not only by commercial (business) and scientific

3.6 PRINTING SERVICES

A broad range of printing options is available for job output from the Computer Center. Output can be listed at an interactive terminal in the user's office, at a convenient remote printer, or at the central facility. If a job produces several different SYSOUT data sets, they can be directed to different locations. The high-speed printers at the central facility offer the greatest variety of printing services.

As a job's printed output is created, it is automatically spooled to a disk by a piece of software called JES2. The output is then printed as soon as an appropriate output device is available. Some jobs which require extensive operator intervention must be written to a tape which is later mounted and printed. This is done via SPOUT (System Peripheral Output UTILITY), which is described in Section 3.6.2 and Section 11.10.

The Computer Center is in the process of expanding its online printing services. Most output which currently requires the use of SPOUT will be handled by the standard service without the need to create a tape.

Some high volume printing applications are not appropriate for computer output and should be handled through normal government printing services.

Directions on how to specify the printing facilities to be used for a single output data set or an entire job are given in Section 6.5.2 "Output Stream Data Set Parameters," Section 6.11 "Output Control Statements," and Section 6.12 "Printing Control Statements."

3.6.1 Standard Printing Services

A. Laser Printers at the Central Facility

Most listings and job output at the central facility are done on the 3800 Printing Subsystem, which is a high-speed laser printer. It produces output which is of better quality than any impact printer. For more information, see Section 11, "Print, Punch, and Micro Output Facilities."

B. Impact Printers at the Central Facility

The slower 1403 impact printers must be used for jobs which require more than 60 lines per page, overprinting with any character except underline, or SPOUT printing service. For more information on online impact printing, see Section 11.

C. Remote Printers

Printed output can be directed to a remote printer. These printers are part of a Remote Job Entry (RJE) workstation which is owned or rented by the user's organization. The services offered by the various models depend on their features (some, for example print all letters in upper case, others print upper and lower case).

For further information on remote printers and how to direct output to them, see Section 5.2 "RJE Workstations" and Section 6.11.2 "/*DDOUT."

3.6.2 Offline Printing Services (SPOUT)

SPOUT (System Peripheral Output Utility) is a processor which handles output services requiring extensive operator intervention. Section 11.10 describes how SPOUT services are obtained in the batch; see the Document Preparation manual for descriptions of the commands WYLBUR users may employ to receive various SPOUT services.

Output to be processed via SPOUT is written onto a tape which is later mounted for printing. This increases turnaround and is more expensive than the standard online printing services. All SPOUT printing is done on the slower impact printers.

A variety of non-standard forms are available. SPOUT also provides photo quality output on impact printers using a mylar (carbon) ribbon. Special density impact printers and the MN10 print train allow printing of special characters, Greek letters, chemical symbols, superscripts, and subscripts. In addition, users may have a reasonable quantity of their own custom-designed forms and cards stocked for printing and punching. Special carriage control tapes can be created to handle forms with nonstandard depths.

Jobs which specify any of the following must be printed via SPOUT:

.MN10 train	.FORMS:
.Non-standard carriage control loop	-user-supplied forms
.DENSITY=8 or 16	-labels
.PHOTO	

4. DATA STORAGE FACILITIES AND SERVICES

4.1 PUBLIC ONLINE DISKS

The NIH Computer Utility has three series of disk packs available to the user: FILE disks, PDS disks, and TMP disks. The functions and uses of each of these are described below.

There are several advantages to using the online disks instead of private mountable disk packs or tapes.

1. There is no delay in the processing of a job while tapes or disks are fetched from the library. The online disks are always mounted and available.
2. Several jobs can read the same data set at the same time. A tape or mountable disk can be used by only one job at a time.
3. Jobs may fit in a higher priority class since fewer mountable volumes are required.
4. There may be a savings in cost for data sets which are frequently accessed. There is a charge in each job step for every tape or private disk used. Storage on the public online disks is charged for by the day, regardless of the amount of use.

See Section 1.6, "Service Charges," for charging details. A weekly report listing each user's data sets stored on online disks and the daily cost is sent to the user's output box.

Sequential data sets on FILE disks that have remained unused for 45 days are migrated to make more space available. Migrated data sets are easily retrieved (see Section 12.6.4). Backup tapes of FILE and PDS disks are made periodically to facilitate the recreation of data that might be lost through catastrophic hardware or software failure or user error.

Computer Center utilities and procedures described in Section 12 of this guide must be used to access data sets and Volume Tables of Contents (VTOCs) on all public online disks. Using unauthorized imported software for this purpose may violate security regulations; (see Section 3.9.2).

4.1.1 FILE Disks - for most data sets

The online FILE disks (FILE00 - FILE54) are used to store TSO and WYLBUR data sets; data sets which must be accessible to programs running in the batch; and user PDS's which require up to 300 tracks. For directions on how to determine which FILE volumes have the most available space, see Section 12.6.8.

The FILE disks are backed up periodically by the Computer Center. Data sets can be retrieved from the backup tapes using WYLBUR's RECOVER command or the procedure described in Section 12.6.3, "Recovering Lost Data Sets."

An automatic data set migration service removes inactive data sets from the FILE disks and copies them to tape. This service is described in Section 4.1.4; details on how to retrieve data sets which have been migrated are given in Section 12.6.4.

When storing data sets on the FILE disks, the following rules should be observed. Illegal data sets (those which do not conform to the rules) will be scratched (and uncataloged) without warning.

1. Data sets larger than 300 tracks cannot be stored on the FILE disks.
2. The names of all data sets stored on the FILE disks must begin with the user's account number and registered set of initials. For example, aaaaiii.MYDATA will be a legal name if "aaaa" is the user's account number and "iii" is the user's set of programmer initials. Data sets created through WYLBUR and TSO are automatically prefixed by the user's account number and initials. Because the names of data sets on the FILE disks cannot be prefixed by NIHD., they cannot be cataloged for batch use.
3. A single data set may not be split between two volumes.

Additional details on how to store data sets are in Section 6.6 "Creating and Using Data Sets."

4.1.2 TMP Disks - for short-term storage

A series of online shared disk packs is available for the temporary storage of permanent data sets. These disks, labeled TMP001 through TMP005, are scratched on a regular schedule, and no automatic backups are made. The scratch schedule is unchanged in spite of holidays unless previously announced.

SCRATCH Schedule:	
Day saved	Deleted before 8:00 a.m.
Monday	Wednesday
Tuesday	Thursday
Wednesday	Friday
Thursday	Saturday
Friday Saturday Sunday	Tuesday

Data sets stored on the temporary storage disks are regulated by the following:

1. Only data sets requiring 300 or fewer tracks are permitted.
2. Normal data set naming conventions must be followed. That is, aaaa.iii.dsname, where "aaaa" is the account number and "iii" is the programmer's set of registered initials. Data sets created through WYLBUR and TSO are automatically prefixed by the user's account number and initials. Because the names of data sets on the FILE disks cannot be prefixed by NIHD., they cannot be cataloged for batch use.
3. Data sets created by TSO are placed, by default, on the TMP disks.
4. WYLBUR can read (USE) and write (SAVE) data sets on these disks. The WYLBUR FIND or SHOW DSNAMEs command may be used to determine what data sets a user has on the TMP

disks and when they were created. The SCRATCH command can be used to remove a data set from a TMP disk.

5. If a data set is needed for longer than it can be kept on a TMP disk, it may be copied onto one of the FILE or PDS disks (see Sections 4.1.1 and 4.1.3) or backed up on tape. Alternatively, the USE and SAVE commands of WYLBUR may be used to place a copy on one of the FILE disks.
6. A single data set may not be split between two volumes.
7. Additional details on how to store data sets are in Section 6.6 "Creating and Using Data Sets."

Data sets which violate the rules will be scratched and uncataloged without warning.

4.1.3 PDS Disks - for libraries

A series of online shared disk packs (PDS001 through PDS011), has been made available for the storage of large Partitioned Data Sets (PDS's).

Data sets stored on the PDS disks are regulated by the following:

1. Only partitioned data sets of 20 tracks or more will be permitted. If a user PDS contains fewer than 20 tracks, it should be stored on a FILE disk.
2. Normal data set naming conventions must be followed. That is, aaaaiii.dsname, where "aaaa" is the account number and "iii" is the programmer's registered set of initials. Because these data set names cannot be prefixed by NIHD., they cannot be entered in the system catalog.
3. WYLBUR and TSO can use the PDS disks. If a data set saved on a PDS disk is to be accessed through TSO, it must first be entered in the TSO catalog. If a data set in the TSO catalog is scratched by a batch job, it must be uncataloged through WYLBUR, TSO or the TSOCAT procedure (described in Section 12.14).
4. A single data set may not be split between two volumes.
5. Additional details on how to store data sets are in Section 6.6, "Creating and Using Data Sets."

Data sets which violate the rules will be scratched and uncataloged without warning.

The PDS disks are backed up periodically by the Computer Center, and data sets can be retrieved from the backup tapes using the procedure described in Section 12.6.3, "Recovering Lost Data Sets." Procedures described in Section 12.6.2, "Saving and Retrieving Disk Data Sets on Tape," can be used for backing up and restoring PDS's.

4.1.4 Data Set Migration and Retrieval

The Computer Center's data set migration system (which currently functions only for the FILE disks) automatically copies inactive disk data sets to tape. Any sequential data set which has been stored for 45 days without being accessed (opened for read or write) will be copied onto a migration tape (and removed from the disk) the next time data sets are migrated. Data sets on migration tapes can be retrieved through WYLBUR or by using the procedure described in Section 12.6.4 "Retrieving Data Sets Migrated from FILE Disks."

The Data Set Migration Facility is carefully designed to provide maximum protection for the migrated data sets. Two identical sets of migration tapes are created and their useability is verified before any data sets are scratched from the disks. Current Computer Center policy is to keep migration tapes permanently.

Each user who has migrated data sets is given a listing of them each time migration is done. For information on this listing and the data set listing which indicates what data sets are ready to be migrated, see Section 4.5 "Storage Reports".

Migrated data sets may be retrieved only if the account/initials combination with which they were saved is still active; for details see Section 1.5.6 "Terminating Use of Services."

If a data set is empty (no program has ever written data into it), it cannot be migrated; however, the attempt to migrate will cause the data set to be scratched.

Permitting data sets to be migrated does mean that they have incurred at least 45 days of unnecessary storage charges; so groups who are particularly concerned with cost reduction should avoid the practice whenever possible. Section 12.6.2 explains how users can copy disk data sets to their own tapes.

4.1.5 Backup and Recovery

The Computer Center backs up each of the FILE and PDS disks to tape twice weekly, before 8:00 a.m. on Wednesday and Saturday. In case of a catastrophic hardware or software failure which destroys one or more of these disk packs, the tapes can be used to recreate all or part of the data. While every effort is made to recover all the data sets from a damaged FILE or PDS disk, a

4.4.6 Getting a Label Put on a Tape

A user can have a machine-readable volume label (tape label) internally recorded on a tape by bringing it to Output Distribution Services and requesting that it be given a six character volume serial number. This number should include the user's three registered initials if possible. The user must inform them of the density at which the tape will be written. The tape will be labeled and returned to the user's output box within 24 hours. The labeling process prevents access to any data that may have been recorded previously on the tape. Tape labeling is a free courtesy service; Computer Center staff members will not submit the newly-labeled tape for access by a batch job.

4.4.7 Preparing a Virgin Tape for Use

A user must have a tape mark put on an unlabeled virgin tape; otherwise, the tape will run off the reel when the operating system attempts to do label checking. To have a tape mark put on a tape, the user brings it to Output Distribution Services and informs them of the density at which the tape is to be written. The tape will be tape-marked and returned to the user's output box within 24 hours.

4.4.8 Having a Tape Cleaned and Degaussed

If the condition of a tape deteriorates so that it cannot be read and the job abends with a 613 or shows repeated I/O errors, the user should, if possible, get a new copy of the tape from the original source. A job using a faulty tape should not be run repeatedly in hopes that it will read properly. It is possible to have such a tape cleaned by Plotting Services, but this procedure is not recommended since data on the tape may be destroyed. For assistance with tape problems, contact the PAL Unit.

The user can have a tape, which is stored in the Information Media Library, cleaned or degaussed (erased) by contacting the tape librarian. If a user has a tape and wants to have it cleaned or degaussed, he can take it himself to Plotting Services.

4.5 STORAGE REPORTS

Three types of storage reports are distributed to users' output boxes. The "Disk Data Set Report" and "Tape/Disk Report" are printed weekly; the "Migration Report" is a cumulative list of each user's migrated data sets, distributed immediately after a migration is done.

4.5.1 Disk Data Set Report

The "Online Disk Data Set Report" provides a list of all data stored on FILE, PDS, or TMP disks under each user's account number and registered initials. Aside from giving the data set names and the volume they reside on, the report lists the following information for each data set: date last used, date last updated, use count, DCB attributes, space taken, and daily and monthly costs. An asterisk next to the VOLUME column indicates that a data set has not been used in the last 30 days.

"LAST USE" indicates the most recent date the data set was opened for input (e.g. USED by WYLBUR, LISTED or EDITED by TSO, or OPENED by TSO or the batch).

"LAST UPDATE" indicates the most recent date the data set was opened for OUTPUT, OUTIN, or UPDATE (e.g. SAVED by WYLBUR, EDIT -- SAVED by TSO, or updated by TSO or a batch job). Link editing into an existing PDS or a random update of an ISAM file will also change the date.

"USE COUNT" reflects the number of times the data set has been opened by WYLBUR, TSO, or the batch since it was first created. The USE COUNT is reset to 1 whenever the data set is recreated (e.g. by WYLBUR's RESAVE command). This field has a maximum value of 32,767; when exceeded, it begins again at one.

Unwanted data sets should be deleted to reduce costs to the user and to free online space.

4.5.2 Migration Report

After each migration, users are sent a cumulative report of all data sets with their account and initials that have been migrated. This list contains the retrieval numbers of the migrated data sets. Directions for returning such data sets to disk are given in Section 12.6.4, "Retrieving Data Sets Migrated from FILE Disks".

Since migrated data sets cannot be scratched, a user's migration report often becomes very long as subsequent migrations are done. Therefore, to remove the unwanted data set names from the active migration index, the user may instruct WYLBUR to FIND a migrated data set and then FORGET it (FIND dsname FORGET).

6. DEFINING AND CONTROLLING A BATCH JOB

This section describes the Job Control Language (JCL) statements needed to describe the resources needed by a batch job running at this installation. Elementary JCL concepts are explained in the manual, NIH Job Control Language (JCL) Primer. Readers of this section are assumed to have a thorough knowledge of that publication. Additional details for any JCL statements may be obtained from the IBM OS/VS2 JCL manual, Form No. GC28-0692.

The IBM System/370 Operating System introduces programs to the computing system, initiates their execution, and provides them with all the resources and services necessary for them to do their work. It is made up of a general library of programs that can be tailored to meet many requirements. The installation can select the systems programs that it needs, add its own programs to them, and update existing programs as needs change.

For illustrative purposes, the programs and routines that compose the operating system are classified as a control program and processing programs. The three main functions of the control program are to accept and schedule jobs in a continuous flow (job management); supervise on a priority basis each unit of work to be done (task management); and simplify retrieval of all data, regardless of the way it is organized and stored (data management). The processing programs consist of language translators (such as the FORTRAN compiler), service programs (such as the Linkage Editor), and problem programs (such as users' programs). The processing programs are used to define the work that the computing system is to do and to simplify program preparation.

Job Control Language (JCL) is the means by which the user communicates with the operating system. Through JCL, the user tells the computer what to do with programs, data sets, and I/O devices. JCL is the only means of getting a batch job on and off the computer; therefore, every user must understand JCL.

JCL is checked in two steps. Conversion processing is done when the job first enters the job queue. During conversion, syntax errors are checked, and all JCL (including expanding procedures) is resolved.

Later, at interpretation, the system detects other errors, including misplaced JCL statements. If a job fails during conversion due to invalid syntax of the JCL and the job is corrected and resubmitted, "logical" errors might still be found during interpretation. Users at terminals will be informed of failures if the jobs in question have the NOTIFY attribute. This is particularly useful for jobs which are run overnight. The syntax errors will be detected, and the job will print immediately or be placed in the OUTPUT HOLD queue giving the user a chance to correct the JCL and resubmit the job. If a job is

directed to OUTPUT HOLD, the output from the conversion phase will be available when that phase is completed. If the job has the NOTIFY attribute, the user will be notified when conversion is completed if he is signed on to WYLBUR or TSO at the time. For a facility to thoroughly check JCL before final job submission, see Section 12.12 "JCL SCANNER."

There are three JCL statements of major concern. They are: JOB, EXEC, and DD. All three are necessary and are discussed below. The DD (data definition) statement is the one which gives the most trouble, and therefore it is discussed in the greatest detail.

The following manuals relating to the IBM System/370 Operating System and to Job Control Language are available from the Technical Information Office:

A. IBM Introduction to Virtual Storage in System/370, Form No. GR20-4260

This student text presents virtual storage concepts and relates them to a conceptual virtual storage operating system. The operating systems and hardware to support virtual storage in System/370 are also described.

B. IBM OS/VS2 JCL, Form No. GC28-0692

This publication describes the facilities provided with JCL and contains the information necessary to code the Job Control Language statements.

C. IBM OS/VS Message Library: VS2 Systems Messages, Form No. GC38-1002

This publication lists, explains, and suggests appropriate responses to the system messages produced by the IBM-supplied components of the operating system.

D. IBM OS/VS Message Library: VS2 System Codes, Form No. GC38-1008

This publication lists the system completion codes and wait state codes for OS/VS2.

E. IBM OS/VS Data Management Services Guide, Form No. GC26-3875

This publication describes the input/output facilities of the operating system. It contains information on record formats, data set organization, access methods, direct access device characteristics, data set disposition, and space allocation.

F. IBM OS/VS Tape Labels, Form No. GC26-3795

This publication describes how the operating system processes magnetic tape volumes. Included are discussions of label formats

they are not used. They should be counted against the totals for these steps.

- d. Scratch tapes used for offline output (e.g. MICRO. or PLOT.) cannot be accessed by the user after the job that creates them is completed.

- (4) Any 7-track tape which is to be kept at job end must be preassigned.

D. Class F

This job class may be specified by the user only when retrieving data sets which have been migrated. See Section 12.6.4 "Retrieving Data Sets Migrated from FILE Disks". It is also used by the retrieval job generated by the ADSRECOV procedure described in Section 12.6.3 "Recovering Lost Data Sets".

6.2 JOB STATEMENT SIMPLIFIED

A simple form of the JOB statement is:

```
//jobname JOB (aaaa,box),name
```

Basic Coding Information

- "jobname" is 3 to 8 alphanumeric or national (\$, #, @) characters. Characters 1-3 must be the programmer's set of registered initials. If several jobs are being submitted, characters 4-8 of the jobname should be unique, otherwise the turnaround time for the jobs may be increased.
- "aaaa" is the programmer's account number.
- "box" is the box number representing the programmer's assigned box in Output Distribution Services. (one to three digits or M, followed by one to three digits).
- "name" is the name of the programmer responsible for the job and contains 1 to 20 characters. If the name contains any characters other than letters, numbers, hyphens or leading or imbedded periods, it must be enclosed in single quotes.

Additional Information

- All of the fields above are required. If the simplest form of the JOB statement is used, the job will be run as class B. This job class permits the use of magnetic tapes and is generally completed in less than two hours. For detailed information on the limitations to class B, see Section 6.1.1 and Section 6.1.3.
- See Section 1.5 "Registering for Use of Services" for information on how to obtain a valid set of initials and an account number.
- If the job does not require the use of tapes, class A may be used if it does not exceed the limits specified in Section 6.1.1 and Section 6.1.3. A class A JOB statement may have the following format:

```
//jobname JOB (aaaa,box,A),name
```

A class A job is usually completed within 30 minutes.

Examples

```
//ABCJOB1 JOB (ACT1,33),SMITH.J
```

```
//XYZ1 JOB (LMNO,M500,A),'MARY JONES'
```


6.3 JOB STATEMENT

The JOB statement indicates to the operating system the beginning point for a job. Every job must have a standard Computer Center JOB statement.

```
//jobname JOB (accounting-information),name,kwd-parms  comments
```

The JOB statement consists of the characters // in columns 1 and 2 and six fields: jobname, JOB, (accounting information), name, kwd-parms, and comments.

Usage at NIH

-- The following parameters cannot be used:

ADDRSPC	NOTIFY	RD
CLASS	PASSWORD	TIME
GROUP	PERFORM	TYPRUN
MSGCLASS	PRTY	USER

Basic Coding Information

- "jobname" is 3 to 8 alphanumeric or national (\$, #, @) characters. The first three characters must be the programmer's registered initials.
- "accounting-information" is a positional parameter.
- "name" contains 1 to 20 characters and identifies the programmer responsible for the job. If the "name" contains any characters other than letters, numbers, hypens, or leading or imbedded periods, must be enclosed in single quotes.
- One or more blanks must be coded following the "jobname," JOB and "kwd-parms" fields.
- The following "kwd-parms" parameters can be used, in any order, after the "name" field:

COND
MSGLEVEL
REGION
RESTART

Examples

```
//XYZPRINT JOB (ABC1,101),ZEBRA
```

```
//ABCSTAT JOB (XYZ2,74,A),'AHAB CAMEL',REGION=300K
```

6.3.1 Accounting Information Parameter

The accounting information lists basic information needed to process any job.

(aaaa,box,cl,tm,lpj,cds)

The parentheses are required.

The format of a simple JOB statement containing all of these items would be:

//jobname JOB (aaaa,box,cl,tm,lpj,cds),name

Usage at NIH

- There are 6 positional subparameters available: the first two subparameters are required, and commas must be supplied for any subparameters that are omitted if further subparameters are to be specified. The subparameters are:
- "aaaa" is the account number and must be supplied.
- "box" is the box number, representing the user's assigned box in Output Distribution Services (one to three digits or M, followed by one to three digits for mailing purposes). It is required.
- "cl" is the class (one letter: A, B, C and F are the only legal classes; B is the default).
- "tm" is the maximum Model 370/168 CPU time in seconds (an integer of one to four digits).
- "lpj" is the maximum number of lines, in thousands, per job (one to four digits).
- "cds" is the maximum number of cards punched, in thousands, (one to three digits).

Additional Information

- The account number supplied is checked with the three initials that begin the jobname. If they are not a valid set, the user's job will be terminated.
- The estimate for CPU time must allow for a possible plus or minus variance of 15 percent between runs of the same job.
- See "Job Class Standards" for specific maximums and defaults for "tm," "lpj," and "cds."
- The time estimate is made to prevent excessive cost if the program gets into a loop. We strongly advise against indiscriminate use of the maximum value of this parameter.
- The maximum number of lines for a job is equal to the number of lines for one copy of output multiplied by the number of copies requested.

6.5.4 DCB Subparameters

A. COMMONLY USED DCB SUBPARAMETERS

Blocksize (BLKSIZE), logical record length (LRECL), and record format (RECFM) are required fields in the data control block for most applications.

DCB=(BLKSIZE=physical-bytes,LRECL=logical-bytes,RECFM=record-format)

Usage at NIH

- Small blocksizes can severely degrade processing efficiency for large data sets.
- Even though large blocksizes will increase job step REGION requirements, an overall reduction in cost will often be obtained from the corresponding reduction in I/O count.
- Blocksizes between 4254 and 6233 are recommended.
- Very large blocksizes hamper tape hardware and software error detection and correction mechanisms.
- Blocksize should be compatible with the type of device on which the data set will be stored.
- Blocksize must be compatible with all software which is to access the data set. (For example, the largest blocksize the Linkage Editor can process is 3200).
- Variable spanned data (RECFM=VS or VBS) which is stored on a FILE pack and is migrated cannot be retrieved if it contains records longer than the specified LRECL.

Basic Coding Information

- "physical-bytes" is the maximum number of bytes in a block (physical record). This value may not exceed 32760.
- If the data set blocksize is not specified by the processing program or on the data set label, the BLKSIZE subparameter must be used.
- "logical-bytes" specifies the actual length, in bytes, for fixed-length records. For variable length records, "logical-bytes" must be 4 bytes larger than the maximum data record within the data set.
- "record-format" specifies the format and characteristics of the data set records. The three basic formats are:
 - F - Records are of fixed length
 - U - Records are of undefined length
 - V - Records are of variable length
- All supported access methods except BDAM allow blocking of fixed length and variable length records. In this case "record-format" would be one of the following:
 - FB - Fixed length records, blocked
 - VB - Variable length records, blocked
- The sequential access methods, BSAM and QSAM, allow "S" to

be specified as a characteristic of fixed length or variable length records. For fixed length records, (RECFM=FBS), this indicates that the records are written as standard blocks (no truncated blocks, except the last). For variable length records (RECFM=VS or RECFM=VBS), this indicates that a logical record may span more than one block.

- The BPAM, BSAM and QSAM access methods allow the RECFM to indicate that the records contain printer control characters. The letter "A" designates ASA control characters. The letter "M" designates machine control characters. The following combinations are valid:

UA	UM	FA	FM
FBA	FBM	FBSA	FBSM
VA	VM	VBA	VBM
VSA	VSM	VSBA	VSBM

- Concatenated data sets must be coded in order by BLKSIZE with the largest first.

Examples

```
//OUTRECS      DD DCB=(RECFM=FB,LRECL=100,BLKSIZE=10000),
//              UNIT=9T1600,VOL=(PRIVATE,SER=654321),
//              DISP=(NEW,PASS),DSN=ABC1XYZ.JULY.REPORT

//GO.FT09F001  DD DCB=(RECFM=VBS,LRECL=9000,BLKSIZE=6000),
//              UNIT=FILE,VOL=SER=TMP004,DISP=(NEW,KEEP),
//              DSN=ABC1XYZ.DATA,SPACE=(TRK,(10,5))
```

- 5 noncontiguous areas (extents), to satisfy the primary quantity; otherwise, the job will be terminated.
- If a secondary quantity is required, there must be enough space in up to 5 noncontiguous areas (extents) to satisfy each request for the secondary quantity. Additionally, no more than 16 noncontiguous areas (extents) will be used to satisfy the entire SPACE request (primary and total secondary). If this space is not available, the job will ABEND.
 - Expansions of existing data sets use the secondary quantity to allocate more space. BDAM data sets cannot be expanded once they are created.
 - Directory space must be allocated when a partitioned data set is created.
 - If the data set is an ISAM data set, the SPACE parameter can specify the number of cylinders to be allocated for an independent index. Computing space requirements for ISAM data sets is complicated; see the IBM Data Management Service Guide, (GC26-3875) for details.
 - If CONTIG is specified and the requested contiguous space is not available, the job will be terminated. Contiguous space should not be requested unless it is required by the program. Requesting it reduces the chance of there being suitable space available. Contiguous space will always be allocated if it is available.
 - If contiguous space is requested and a secondary quantity is coded, the primary request will be satisfied with contiguous space, but the secondary quantity may not be contiguous.
 - The procedure PUBLIST, described in Section 12.6.8, may be used to make certain that space is available on a particular disk.

Examples

```
//STEP1      DD SPACE=(TRK,(10,1)),UNIT=SYSDA,
//           DISP=(NEW,PASS)

//FIRST      DD SPACE=(3120,(100,10),RLSE),UNIT=FILE,
//           DSN=ABC1XYZ.FORX.STATS,DISP=(NEW,KEEP),
//           DCB=(RECFM=FB,LRECL=80,BLKSIZE=3120)

//ONE        DD SPACE=(CYL,(20,,1),,CONTIG),UNIT=SYSDA,
//           DCB=(RECFM=FB,LRECL=80,BLKSIZE=3120,
//           DSORG=IS),DISP=(NEW,PASS)
```


- 2) The data set is cataloged, and the VOLUME parameter is not specified.
 - 3) VOL=REF=reference is coded.
 - 4) The data set has SYSOUT=A, B, or C specified.
 - 5) DUMMY, *, or DATA is used to define the data set.
- If a data set is passed from a previous step, the UNIT and VOLUME parameters should not be specified.
- If the data set is cataloged and the VOLUME parameter is not coded, the system will obtain unit and volume information from the catalog. If UNIT is specified, it must be compatible with the catalog information and valid for the data set. However, if the VOLUME parameter is coded, UNIT must be supplied because the catalog will not be searched.
- If VOL=REF=reference is coded (see Section 6.5.14), the data set uses the same volume assigned to an earlier data set. The system obtains unit and volume information from the earlier DD statement.
- If UNIT=DEFER is specified and the data set on that volume is never opened, the volume is not mounted even though the UNIT is allocated. However, this will save an unnecessary tape mount if the tape may not be needed depending on what happens in other steps. No other jobs can use that volume until the step specifying DEFER ends, and any attempt to use such a volume by other jobs will cause all but the first job to enter a "wait" state.
- It is strongly recommended that multi-volume data sets use only one drive, rather than specifying "count" or "P" to use multiple drives. The use of these parameters will not generally affect the execution time of an individual job; however, it may degrade performance of other jobs in the system.
- Two data sets on different volumes can be assigned to the same device by specifying UNIT=AFF=ddname, where "ddname" defines the DD statement of the data set to share the drive.
- A job specifying a tape other than a 9-track, 1600 BPI tape (UNIT=9T1600), must include a /*ROUTE XEQ resource statement (see Section 6.9.2).

Examples

```
//SORTWK01      DD UNIT=SYSDA,SPACE=(CYL,(80))

//GO.FT10F001   DD UNIT=(9T1600,,DEFER),DISP=(OLD,KEEP),
//              VOL=(PRIVATE,SER=123456),DSN=ABC1XYZ.OUTPUT,
//              LABEL=(,,IN)

//INPUT         DD UNIT=FILE,VOL=SER=TMP001,DISP=SHR,
//              DSN=ABC1XYZ.INPUT
```

6.5.14 VOLUME Parameter

The volume parameter provides information about the tape or disk volume(s) on which a data set resides.

```
VOLUME (PRIVATE,RETAIN,seqno,count,SER=(serial,serial...))
or
VOL      REF=dsname
          REF=*.ddname
          REF=*.step.ddname
          REF=*.step.procstep.ddname
```

Usage at NIH

- All NIH Computer Center tapes have 6 numeric digits as volume serial numbers; zeroes should be inserted to the left to pad to 6 digits if necessary.
- All NIH Computer Center mountable disks have a letter "D" followed by 5 numeric digits.

Basic Coding Information

- "PRIVATE" indicates that output data sets cannot be allocated to this volume unless the volume is specifically requested. The volume will be demounted after its last use in the job step, unless RETAIN is coded or a data set on the volume is passed.
- "RETAIN" indicates that the volume is not to be demounted after its last use in the job step. The volume will be kept mounted until its next reference, or job end, whichever comes first.
- "seqno" indicates which volume of an existing multi-volume data set is to be processed first.
- "count" indicates the maximum number of tape volumes an output data set may require. This must be specified if the data set is to exceed 5 volumes and the specific volume serial numbers are not coded.
- "SER=(serial,serial,serial...)" gives the serial number(s) of the volume(s) on which the data set resides; they must be in order of creation.
- "REF" indicates that the serial numbers of the volumes are identified in an earlier DD statement in the job or in the catalog.
- "REF=dsname" tells the system to use the same volumes that were assigned to the named passed or cataloged data set.
- "REF=*.ddname" tells the system to use the same volume(s) that were assigned to the named DD statement, which must precede this statement in this job step.
- "REF=*.step.ddname" tells the system to use the same volume(s) that were assigned to the named DD statement which appears in an earlier step named "step" in the same job.
- "REF=*.step.procstep.ddname" tells the system to use the same volume(s) assigned to the named DD statement in a

cataloged procedure. "step" is the name of the job step that executes the procedure. "procstep" is the name of the procedure step that includes the named DD statement, and "ddname" is the name of the DD statement that contains the volume serial number(s).

-- The VOLUME parameter must be given:

- 1) To create a new data set on a specific volume
- 2) To create a new data set on more than five scratch volumes
- 3) To request an old data set which is not cataloged or passed
- 4) To mount a "scratch" volume which the user wishes to have demounted and kept at job end as a "private" tape

-- If the only subparameter coded is PRIVATE, SER=, or REF=, the parentheses are not required.

-- If only one serial number is coded, it need not be enclosed in parentheses. All subparameters other than SER are positional; therefore, omitted subparameters that are followed by other subparameters must be replaced by a comma. SER is a keyword subparameter and must follow all positional parameters.

Additional Information

-- If a volume is to be PRIVATE, every DD statement referencing that volume should have PRIVATE coded.

-- Existing data sets must have a specific volume request. A specific volume request is implied when:

- 1) VOL=SER=serial is coded on the DD statement.
- 2) The data set is passed from an earlier job step or is cataloged.
- 3) VOL=REF=reference is coded on the DD statement, referring to an earlier specific volume request.

-- New data sets may have a specific volume request or allow the system to assign a scratch volume.

-- If scratch volumes assigned by the system to a job are to be kept after the job, the job must be run Class C (see Section 6.1.4).

-- If UNIT=FILE is specified, and no specific volume request is made, the system will search the TMP disk packs to locate one which will satisfy the primary space request and does not contain a data set with the same data set name.

-- When using a passed or cataloged data set, do not code any serial numbers or use VOL=REF=reference. Doing so would mean the data set would not be referenced through the system's internal pass table or the catalog.

-- If a multivolume data set is to be MODed in more than one step, the user must be aware of which tapes are written on in each step and specify the sequence number for the tape on which the data set ends.

Examples

```

//GO.PRINT1      DD VOL=PRIVATE,UNIT=9T1600,
//              DSN=ABC1XYZ.PRINT,DISP=(NEW,KEEP)

//GO.FT10F001    DD VOL=(PRIVATE,RETAIN,SER=(123456,012345)),
//              UNIT=9T1600,DSN=ABC1XYZ.DATA,LABEL=(,,IN),
//              DISP=(OLD,KEEP)

//INPUT          DD VOL=SER=FILE00,UNIT=FILE,
//              DSN=ABC1XYZ.INPUT,DISP=SHR

//GO.FT11F001    DD VOL=(PRIVATE,REF=*.GO.PRINT1),UNIT=9T1600,
//              DSN=ABC1XYZ.DATA2,DISP=(OLD,KEEP),LABEL=(2,,,IN)

//GO.EZFOUTPT    DD DSN=MICRO105.XYZ1ABC.FISH
//              DISP=(MOD,KEEP,DELETE),VOL=(PRIVATE,,2)
//              DCB=(RECFM=FBA,LRECL=135,BLKSIZE=3915)

```

6.5.15 SYSUDUMP and SYSABEND Statements

Every program which is debugged and running on a production basis should contain a //SYSUDUMP or //SYSABEND statement. If the program executes properly, inclusion of the dump statement costs nothing. If the program terminates abnormally (ABENDs), more often than not a dump is indispensable for determining the cause of the problem. Any job step which is subject to ABEND should contain a DD statement of the following format:

```

//SYSUDUMP DD SYSOUT=A
//SYSABEND DD SYSOUT=A

```

The SYSUDUMP DD statement provides a dump of the user program area while the SYSABEND DD statement includes the system nucleus along with the problem program area. If this statement is not in the job, only a one line completion code indication will be given. If a job "hangs up in a wait state", the operator will cancel the job and attempt to get a dump. Again, one of these DD statements must be present. I/O bound jobs which are run on a production basis should use SYSABEND DD statements instead of SYSUDUMP statements. Many of the errors which occur in this type of job can be diagnosed only with a SYSABEND dump, which means it is required before the problem can be found, or a refund given.

The /*CNTL statements are typically used to regulate the execution of jobs which access a private resource shared by many users. The resource may be a tape containing one or more data sets, a disk data set, or a private disk. A /*CNTL statement with the same controlname is included in each job which accesses the resource. Jobs which may share the resource (e.g., reading a program library) contain a /*CNTL controlname,SHR statement. Jobs which must have exclusive use of the resource (e.g., updating a program library or using a tape or private mountable disk) contain a /*CNTL controlname,EXC statement.

Please note that if access to a resource is to be controlled, all jobs accessing it must contain a /*CNTL statement with the same controlname.

If multiple jobs with the same /*CNTL EXC statement are awaiting execution at the same time, they will be executed in strict job class order (i.e. class A jobs will execute before class B). If a class A job is set to level 0 (so it will be held and not run), all /*CNTL EXC class B and C jobs with that controlname will also be held.

Since /*AFTER and /*BEFORE cannot be used in combination with /*CNTL, the user may wish to use DSSUBMIT to control the order of execution. (DSSUBMIT is described in Section 12.11.)

6.9 RESOURCE (TAPE OR DISK) CONTROL STATEMENTS

6.9.1 /*MESSAGE

If a job requires the mounting of any preassigned tapes or disks, it must include the following "message" statement immediately following the JOB statement:

```
/*MESSAGE xxxxxx,v;yyyyyy,v
```

where "xxxxxx" and "yyyyyy" are the volume serial numbers of the tapes or disks requested; v may be replaced with an R to allow reading only or a W to permit reading or writing; R is the default. There must be at least one blank after the word MESSAGE. The message portion of the message statement cannot extend beyond column 72. Although message statements may not be continued, as many as are necessary may be used. If a private tape or disk is requested by a job and no message statement has been used to convey this information, the job will ABEND. Remember no message statement is needed for online disks since they are already mounted.

Users who are processing tapes that are not part of Computer Center's tape library, (for example, tapes acquired from other installations or tapes used to record laboratory data), should handle these tapes as follows:

- (1) Give the tape to Output Distribution Services before the job is submitted.
- (2) Include the following message statement in the job:

```
/*MESSAGE volser,vS
```

where "volser" is the exact volume serial number (this must be the same as that on the external label on the tape case).

This tells the operators that the tape called for by that job is a special user tape and will come from Output Distribution Services rather than the tape library. For further information, see Section 4.4.4 "Exchanging Tapes with Other Installations".

User comments may be included by leaving at least one space after the last volume element and enclosing the comment within single quotes. This facility is for user documentation only and may not be used for operator communication.

Examples

```
/*MESSAGE MYTAPE,'S
```

6.10 JOB EXECUTION CONTROL STATEMENTS

6.10.1 /*KEYWORD

The keyword facility protects the user's account and initials against unauthorized use. The following card must be included or the job will not run. It need not be supplied for jobs submitted through WYLBUR or TSO since both automatically supply it.

```
/*KEYWORD=kkk
```

where "kkk" is the keyword.

When the JCL for the job is printed on the job output, the keyword will be replaced with blanks. Additional information on keywords, is given in the WYLBUR Fundamentals and General Editing manuals, the NIH TSO Reference Manual, and Section 12.18 "Setting a Keyword from a Batch Job" and Section 1.5 "Registering for Use of Services".

6.10.2 /*NOTIFY

The NOTIFY facility allows a user who is signed on to WYLBUR or TSO to be notified when his batch job reaches completion. Inserting a

```
/*NOTIFY
```

card into any batch job causes a message to be sent to both TSO and WYLBUR when it completes execution and again when its output is completed. The message is sent to the initials on the job statement provided the user is logged on to WYLBUR or TSO at the time; if he is not logged on, the message is not sent.

```
/*NOTIFY iii
```

can be used to send the message to the specified initials. Only one user can be notified; so, if this option is used, the messages will not be sent to the initials on the JOB statement.

6.10.3 /*DISCOUNT

The DISCOUNT service allows users to receive a 50 percent rate reduction on batch processing for all job classes running between 6 p.m. and 6 a.m. weekdays and all day weekends. To request the DISCOUNT facility include a

```
/*DISCOUNT
```

statement after the JOB statement along with any other JES2 control statements needed or include DISCOUNT as an operand in a

WYLBUR RUN, LIST, PUNCH, or RETRIEVE command or a TSO LISTOFF or SUBMIT command. However, unless a LIST command generates a SPOUT tape, no job is submitted to the system for execution and DISCOUNT is not applicable. If a SPOUT tape is created, the discount rate applies to the job creating the SPOUT tape.

Any job containing a /*DISCOUNT statement will automatically be held for execution until the next discount period begins. For example, if a job which specifies DISCOUNT is submitted during the day (non-discount period), it will be held and automatically released to the system at 6 p.m. If a job with DISCOUNT specified is not processed by 6 a.m. the following day, it will be held until the next evening, unless NODISCOUNT is specified, through WYLBUR or with the RJE workstation command described in Section 5.2.9.

Even though a discount job will be executed only during the discount period, all mountable volumes (e.g. special tapes) must be at the Computer Center when the job is submitted.

Even if a job does not include a /*DISCOUNT statement, any job steps which begin execution during the discount period will receive the discount rate.

6.10.4 /*QUICK

The QUICK turnaround facility is used to obtain better service for batch jobs submitted during the discount period. This service was established to enable a user who is waiting for output during the evening hours to bypass the long nighttime production runs. The facility is activated by the inclusion of a

/*QUICK

card in the job or by specifying QUICK as an operand on the RUN, LIST, PUNCH, and RETRIEVE commands (WYLBUR) or the SUBMIT and LISTOFF commands (TSO). QUICK can be specified only on jobs being submitted during the 6 p.m. to 6 a.m. time period. If QUICK is specified at any other time, the user will be sent a message, and the job will be run as a regular (non-QUICK, non-DISCOUNT) job of its class. QUICK should not be used for a priority level 0 job.

6.10.5 /*RERUN

If a job should be rerun in the event of a system crash, the following statement should be included:

/*RERUN

6.12 PRINTING CONTROL STATEMENTS

For an overview of the printing services offered at the NIH Computer Utility, see Section 3.6.

6.12.1 /*IMPACT

Jobs requiring the use of an impact printer instead of a laser printer must include the statement:

```
/*IMPACT
```

Impact printers are needed for all types of overprinting except for underscores. Three common types of overprinting are: 1) boldface (where the same character is overprinted several times); 2) "masking" a field (by overprinting with different characters), and 3) building composite characters (by combining two or more characters).

If a /*IMPACT statement is included in a job, all job output (including SYSOUT data sets) will be printed on impact printers.

6.12.2 /*LINES

The frequency of automatic page ejects can be set with the following statement:

```
/*LINES=n
```

where "n" is the number of lines per page, (1-255).

If "n" is more than 60 the job will be printed on an impact printer. The default is 60 lines per page. If both /*PP and /*LINES are included, only the last will be recognized.

6.12.3 /*NOEJECT

Normally page ejects (controlled by JES2) are issued every 60 lines. To suppress this, include a

```
/*NOEJECT
```

statement. If this statement is included, page ejects will be issued only when the user so indicates, via carriage control in his program. Any job with this option will be printed on an impact printer. /*PP has the same meaning. If both /*NOEJECT and /*LINES are included, only the last will be recognized.

6.12.4 /*UNN and /*NUM

A user submitting a job through WYLBUR can suppress the line numbers, which will automatically be printed in columns 73/80 of each line which has blanks in those columns, by including the following statement:

```
/*UNN
```

Including a /*NUM statement in any job causes line numbers to be written over the contents of columns 73/80.

```
/*NUM
```

/*NUM and /*UNN statements may be included anywhere in the range to be run; they have the effect of enabling or inhibiting line numbering until the end of the range or until the next /*NUM or /*UNN statement is encountered. The statements themselves are never submitted for processing.

6.12.5 /*TITLE

To print a title on the header and trailer pages of a batch job listing, include the following statement:

```
/*TITLE=title
```

Where "title" is up to 20 characters beginning with the first non-blank character after column 8.

If more than 20 characters are given in the title, only the first 20 are used. The equal sign is optional; a blank may be substituted. If more than one /*TITLE statement is present, the last encountered will be used.

Examples

```
/*TITLE MASTER LISTING
```

```
/*TITLE=July 8, 1980
```

```
/*TITLE      Building 12 - 2235
```

7.1 PUBLICATIONS

The following manuals are available through the Technical Information Office:

A. IBM OS/VS-DOS/VS-VM/370 Assembler Language, Form No. GC33-4010

This publication is a reference manual for the programmer using the Assembler language. Part I describes the structure of Assembly Language and its coding rules. Part II describes the machine instruction types and their formats. Part III describes the macro facility and the conditional Assembly language.

B. IBM OS/VS-VM/370 Assembler Programmer's Guide, Form No. GC33-4021

This publication complements the IBM OS/VS and DOS/VS Assembler Language manual. It provides a guide to program assembling, linkage editing, executing, interpreting listings, and Assembler programming considerations.

C. IBM System/370 Principles of Operation, Form. No. GA22-7000

This manual is the machine reference manual for the IBM System/370. It is written principally for Assembler language programmers to be used in conjunction with the appropriate Assembler language manuals.

D. IBM OS/VS2 Supervisor Services and Macro Instructions, Form No. GC28-0683

This publication shows applications programmers how to use the services of the supervisor, the macro instructions used to request these services, and the linkage conventions used by the control program to provide these services.

E. IBM OS/VS Data Management Macro Instructions, Form No. GC26-3873

This publication contains descriptions and definitions for the data management macro instructions available in Assembly Language. No information is included for VSAM.

F. IBM OS/VS Data Management Services Guide, Form No. GC26-3783

This publication describes the data management facilities available under OS/MVS except for VSAM and any specialized applications such as TSO, graphics, optical readers, etc. It contains information on record formats, data set organization, access methods, direct access device characteristics, data set disposition, and space allocation.

G. IBM OS/VS System Programming Library: Debugging Handbook Volume 1, Form No. GC28-0708; Volume 2, Form No. GC28-0709; Volume 3, GC28-0710

This handbook provides reference information for use in debugging user or system programs. The user should have a working knowledge of OS/VS2 functions and logic. The three volumes of the handbook omit many general reference topics included in the System/370 Reference Summary card.

H. IBM OS/VS2 JCL, Form No. GC28-0692

This publication contains the information necessary to code Job Control Language (JCL), JES2 control statements and JES3 control statements. The user should understand the concepts of job management and data management.

I. IBM OS/VS Message Library: VS2 System Messages, Form No. GC38-1002

This publication lists the systems messages produced by IBM-supplied components of the operating system. The causes of the messages are explained, the accompanying actions by the operating system are described, and appropriate responses are suggested.

J. IBM OS/VS Message Library: VS2 System Codes, Form No. GC38-1008

This publication lists the system completion codes and wait state codes for OS/VS2. Each code is explained, and where appropriate, a programmer response is indicated.

K. Assembler (G) User's Guide

This manual describes the capabilities and use of the G-level Assembler which was developed from Assembler-F by the University of Waterloo.

L. INTERFACE

This is a series of technical notes for users, published by the Computer Center.

8.1 PUBLICATIONS

The following manuals about COBOL are available through the Technical Information Office:

- A. IBM OS/VS COBOL Compiler and Libraries General Information,
Form No. GC28-6470

This publication contains brief descriptions of the OS/VS COBOL compiler and library features, information on compatibility, and system requirements.

- B. IBM VS COBOL for OS/VS, Form No. GC26-3857

This publication is a reference manual to be used when writing OS/VS COBOL programs. It gives the rules for writing COBOL source programs.

- C. IBM OS/VS COBOL Compiler and Libraries Programmer's Guide,
Form No. SC28-6483

This publication describes how to compile, link-edit, and execute or compile and load OS/VS COBOL programs.

- D. RHB Routines

Although most of the RHB routines were written for FORTRAN programmers, there are routines in this book designed for the COBOL programmer. Of these, the "COBOL ISAM ERROR ROUTINES", RHB246, and RHB246A, are of special interest. These routines are called from the DECLARATIVES SECTION to process errors on ISAM files and should be used by every COBOL program which processes an ISAM file.

- E. INTERFACE

This is a series of technical notes for users, published by the Computer Center.

8.2 COMPILER OPTIONS

The standard options generated for the COBOL compiler are listed below.

Default Options

- | | |
|------------|--|
| APOS | - Indicates to the compiler that the apostrophe (') is acceptable as the character to delineate literals, and to use that character in the generation of figurative constants. |
| BUF=50K | - Specifies the amount of the SIZE parameter to be reserved for compiler data set buffers. |
| FLAGW | - Print all warning diagnostics as well as error diagnostics. |
| L120 | - The length of each line of reformatted listing is to be 120 characters long. |
| LANGVL(2) | - The 1974 American National Standard COBOL definition is to be used when compiling those source elements whose meaning has changed. |
| LCOL1 | - The Procedure Division part of the listing is to be in single column format. |
| LINECNT=57 | - Number of lines printed per page of source listing. |
| NOADV | - The first byte must be reserved for the carriage control character for files with WRITE ...ADVANCING. |
| NOBATCH | - Multiple programs and/or subprograms are not to be compiled with a single invocation of the compiler. |
| NOCDECK | - COPY statements are not to be expanded into COPY members in the SYSPUNCH output. |
| NOCLIST | - Condensed listing of PROCEDURE DIVISION is not to be listed. |
| NOCOUNT | - Code is to not be generated to produce verb execution summaries at the end of problem program execution. |
| NODECK | - No object deck will be punched. |
| NODMAP | - Glossary and global tables are not to be printed. |
| NODUMP | - The compiler should produce an informative |

9.6 PROGRAMMING AND RUNNING TIPS

For information on inter-language communications, see Section 8.6.6.

9.6.1 Coding in FORTRAN

1. The names used in the SUBROUTINE and the CALL statements referencing a subroutine must always be the same. If this name is not the same as the name under which the subroutine is stored in the online library, then an INCLUDE control statement must also be used for the subroutine in the link-edit step.
2. When using tapes for input, LABEL=(,,,IN) should be coded on the DD statement. If the subparameter, IN, is not coded, the job may be cancelled by the operator. FORTRAN opens input data sets for both input and output. If the tape is mounted for read only and LABEL=(,,,IN) is not coded on the DD statement, the system requires a reply from the operator for each data set that is opened.
3. The FORTRAN-G1 compiler can process approximately 300-400 source statements per compilation depending on the complexity of the statements. The FORTRAN (H Extended) compiler, when used with OPT(1) or OPT(2), generally produces more concise object code and shorter execution time than FORTRAN-G1. However, this optimization causes an increase in compilation time. The FORTRAN (H Extended) compiler can process 200-300 source statements, again, depending on the complexity of the statements. In determining the number of source statements per compilation, each program or subroutine in a batch compile is considered to be a separate compilation. Generally more statements can be compiled by increasing the REGION parameter for the compiler.
4. The FORTRAN debug facility available with the FORTRAN-G1 compiler can be used when debugging a program. It can be used to check the validity of subscripts, trace the flow of the program, output the variables whenever their values change, and output given variables at any point in the program.
5. When using FORTRAN to create variable length spanned records (all unformatted I/O in FORTRAN uses VS or VBS records), the LRECL specified must be as long as the longest record. No error messages will be printed, but the file may not be useable by other programs.

9.6.2 Using Data Set Reference Numbers

1. For the FORTRAN compilers, the data set reference numbers

may range from 1-32 (i.e., FT01F001-FT32F001). The data set reference number nn (FTnnF001) corresponds to the READ (nn,formatno)...etc. in the user's FORTRAN programs.

2. FORTRAN-G1 and H-Extended load module execution time messages are printed on data set reference number 15 (FT15F001). FT15F001 is opened for output by FORTRAN because it is the default ddname for all IHO type diagnostics and execution time messages. The DD statement for unit 15 should always be coded as SYSOUT=A, e.g.

```
//FT15F001 DD SYSOUT=A
```

Never use unit 15 for input; this is an error that will have unpredictable results.

3. The Fortran IV additional facilities, which permit I/O operations without data set reference numbers, are defined by the Computer Center such that:

```

READ  format,list - accesses data set reference number 1
                      (FT01F001 or SYSIN)
PRINT format,list - writes data set reference number 15
                      (FT15F001)
PUNCH format,list - writes data set reference number 16
                      (FT16F001)

```

9.6.3 Writing Efficient Programs

1. Avoid using subroutines and functions for small repeated tasks. Use statement functions for short functions instead of calling external subroutines. Statement functions are compiled directly in the module.
2. Apply the information gained during analysis of the problem before programming to simplify the problem and speed up the numerical procedure.
3. Arrange the program logic to avoid branches whenever possible.
4. Make the most probable result of all logical IF statements a simple drop-through instead of a branch.
5. Reduce I/O to the minimum necessary when debugging is complete.
6. Using IF statements to determine conditional branches to more than three labels is less efficient than using unconditional or computed GO TOs. The computed GO TO uses more overhead time and space than the unconditional GO TO but it is still more efficient than a set of IF statements.

7. Because of round-off error in low-order bits, do not test for equal using floating-point variables; use .GE. or .LE.
8. Use SQRT instead of **.5. For small powers, use A*A*A ... or A**I with I = R instead of A**R where R is a floating-point integer. (Values raised to integer powers are computed by repetitive multiplication, whereas values raised to real powers are computed using logarithm and exponential routines. A**R is about four times slower than A**I).
9. Use logical IF's instead of arithmetic IF's when possible.
10. Calculate all quantities which are constant throughout a program at the beginning, and calculate all quantities constant throughout a loop outside the loop. For example:

```
DO 20 I = 1,450
20  C(I+3,2*I+1) = (D*(I+2))**(2*K)+E-2.*L-1
```

should be written

```
M = 2*K
F = E-2.*L-1
DO 20 J=3,452
20  C(J+1,2*J-3) = (D*J)**M+F
```

If several tests or operations are to be done in DO loops, incorporate as many as possible in a single DO loop rather than setting up separate DO loops for each test or operation.

Use a minimum of mixed-mode arithmetic. The extra coding generated can in some cases take more time to execute than the arithmetic itself. Use X=0.0 instead of X=0; I=0 instead of I=0.0. Choose variable types to avoid conversion whenever possible.

11. Where possible pass variables to SUBROUTINES through COMMON instead of using parameter lists; this saves much time because addresses do not have to be passed down to the subroutine for the variables in the calling sequence.
12. Use the DATA statement or BLOCK DATA subroutine feature for initializing constants.

9.6.4 Increasing I/O Efficiency

1. Use unformatted I/O data files whenever possible.
2. When all elements of an array are to be read or written, use an implied DO loop rather than an actual DO loop. For

example, if you have DIMENSION A(100), use WRITE (6,100)A rather than WRITE (6,100) (A(I),I=1,100).

In the second statement the I/O subroutine IBCOM will be entered once for each item in the array. In the first statement, since A is given without indexing, IBCOM will be entered only once, resulting in a saving of CPU time.

3. If your program reads or writes portions of arrays, the use of EQUIVALENCE statements will permit the use of simple array names in I/O statements. For example:

```
INTEGER BUFFER(1000),BUFF1(500),BUFF2(500)
EQUIVALENCE (BUFFER(1),BUFF1(1)),
*(BUFFER(501),BUFF2(1))
1 WRITE(3)INUM,(BUFFER(I),I=1,500)
2 WRITE(3)INUM,BUFF1
RETURN
END
```

WRITE statements 1 and 2 are equivalent in effect, but the second requires less CPU time.

4. The I/O of a list of many variables can be made more efficient in the following manner. Use a COMMON or named COMMON statement to place the variables in contiguous storage spaces. Define a singly dimensioned array the length of the list of variables and use an EQUIVALENCE statement to cause the array to reference the same storage locations as the list of variables. Then use the array name for I/O. An example is:

```
COMMON/LIST/A(20),B(30),C,D,E(15),F,G(25)
DIMENSION XOUT(93)
EQUIVALENCE (A(1),XOUT(1))
READ (5,100) XOUT
.
.
WRITE (6,200) XOUT
```

F. IBM OS PL/I Checkout Compiler: TSO User's Guide, Form No. SC33-0033

This manual explains, for TSO users, how to invoke the OS PL/I Checkout Compiler and use its conversational capabilities. It indicates what TSO facilities can be used when processing a PL/I program and includes a description of the PLIC command and its subcommands.

G. A PL/I Primer, Form No. SC28-6808

This IBM manual is useful for novice programmers who wish to learn PL/I or for more experienced programmers who want a quick overview of the language. It does not contain a complete description of the language. The material is well presented and includes numerous examples.

H. INTERFACE

This is a series of technical notes for users, published by the Computer Center.

10.2 COMPILER OPTIONS

The standard batch compile-time options generated for the PL/I Optimizer are:

- | | |
|--------------------|--|
| AGGREGATE | - Aggregate length table, giving lengths of all arrays and major structures, will be produced. |
| ATTRIBUTES(FULL) | - Attributes of all source-program identifiers will be listed, including unreferenced identifiers. |
| CHARSET(60,EBCDIC) | - Source-program is written in EBCDIC, using the 60-character set. |
| COMPILE | - Source-program will be compiled unless an unrecoverable error is detected. |
| FLAG(I) | - List all compiler messages, including informatory. |
| GOSTMT | - Source-program statement numbers will be included in execution-time messages. |
| LINECOUNT(60) | - Each page of compiler listing will contain 60 lines. |
| LMESSAGE | - The long form of compiler messages will be produced. |
| MARGINS(2,72,1) | - Columns 2 to 72 available for use in source module. Column 1 may be used for ANS printer control characters, to control spacing of source-program listing. |
| NEST | - Block-level and DO-group level will be indicated for each statement in the source program listing. |
| NOOPTIMIZE | - Specifies fast compilation, but inhibits optimization for faster execution and reduced main storage requirements. |
| OBJECT | - Object module produced by the compiler will be stored in the data set defined by the DD statement named SYSLIN. |
| OPTIONS | - Compiler options used during compilation will be listed. |

2/27/81

i

1. ORIENTATION.....	5
1.1 ORGANIZATION.....	5
1.1.1 DCRT (Division of Computer Research and Technology)	5
1.1.2 Computer Center Branch (CCB).....	6
1.1.3 Data Management Branch (DMB).....	6
1.1.4 Computer Systems Laboratory (CSL).....	7
1.1.5 Laboratory of Applied Studies (LAS).....	7
1.1.6 Physical Sciences Laboratory (PSL).....	8
1.1.7 Laboratory of Statistical and Mathematical Methodology (LSM).....	8
1.2 LOCATION.....	9
1.2.1 NIH Campus.....	9
1.2.2 Building 12.....	10
1.2.3 Building 12A.....	12
1.2.4 Building 12B.....	14
1.2.5 Restricted Access Areas.....	16
1.3 TELEPHONE NUMBERS.....	17
1.3.1 Personnel Directory.....	18
1.3.2 Service Directory.....	20
1.4 OPERATING HOURS.....	22
1.5 REGISTERING FOR USE OF SERVICES.....	23
1.5.1 Registered Initials.....	23
1.5.2 Keyword Protection.....	24
1.5.3 Output Boxes.....	25
1.5.4 Account Authorization Forms.....	25
1.5.5 Updating Registration Information.....	31
1.5.6 Terminating Use of Services.....	31
1.6 SERVICE CHARGES.....	34
1.6.1 Batch Charging Formula.....	34
1.6.2 Discount Processing.....	35
1.6.3 TSO (Time Sharing Option).....	35
1.6.4 WYLBUR.....	35
1.6.5 Offline Output Facilities.....	36
1.6.6 Tape and Disk Storage.....	37
1.6.7 Peripheral Services.....	37
1.6.8 Billing Information.....	38
1.6.9 Refund Policy.....	41
1.7 CONTRACTING GUIDELINES.....	43
2. SERVICES FOR USERS.....	47
2.1 CONSULTING.....	48
2.1.1 Tutorial Assistance - Training Unit.....	48
2.1.2 Programming Assistance - PAL Unit.....	48
2.1.2 Using the Programmer Trouble Report (PTR).....	48
2.1.2 Documenting the PTR.....	49
2.1.2 Submitting PTR's from Terminals.....	51
2.1.2 Obtaining Refunds (PTR).....	52
2.1.3 Terminal Problems - Dedicated Equipment Services...	54

2.2	TRAINING.....	55
2.2.1	Computer Training Courses and Seminars.....	55
2.2.2	Self-Instruction Courses.....	55
2.3	DOCUMENTATION.....	57
2.3.1	Computer Center Publications.....	57
2.3.2	Other DCRT Publications.....	58
2.3.3	Vendor-Supplied Publications.....	59
2.4	USERS' WORK AREA AT THE COMPUTER CENTER.....	60
2.5	DATA CONVERSION.....	61
2.5.1	Data Entry.....	61
2.5.2	Plotting.....	62
2.5.3	Output Handling.....	62
2.6	OUTPUT DISTRIBUTION.....	63
2.7	TEST JOB STREAM.....	64
3.	SOFTWARE FACILITIES.....	67
3.1	OPERATING SYSTEM/370.....	67
3.2	WYLBUR.....	68
3.3	TSO.....	71
3.3.1	TSO/WYLBUR.....	73
3.4	BATCH LANGUAGES.....	74
3.5	UTILITIES.....	76
3.6	PRINTING SERVICES.....	77
3.6.1	Standard Printing Services.....	77
3.6.2	Offline Printing Services (SPOUT).....	78
3.7	GRAPHICS SOFTWARE.....	79
3.7.1	CalComp Subroutines.....	79
3.7.2	Integrated Plotting Package (IPP).....	79
3.7.3	Publication Quality Plots.....	80
3.8	MICROFICHE SERVICES.....	81
3.9	RESTRICTIONS.....	82
3.9.1	Implementation of Non-NIH Software.....	82
3.9.2	Software Features Banned at NIH.....	83
3.10	IMS (Information Management System).....	84
4.	DATA STORAGE FACILITIES AND SERVICES.....	87
4.1	PUBLIC ONLINE DISKS.....	87
4.1.1	FILE Disks - for most data sets.....	87

11.4.3 Print Trains for Offline Impact Printing

The MN10 print train may be used for offline impact printing (via SPOUT) in addition to the trains available for online impact printing (the SN10, CN10, and TN10).

The MN10 train contains many Greek letters and chemical and mathematical symbols. A listing of the characters is given in Section 11.4.4. The WYLBUR Document Formatting manual describes techniques for listing these characters from WYLBUR.

11.4.4 Hexadecimal Chart for Character Sets and Print Trains

HEX	MN10	HEX	TN10	HEX	ON10	HEX	SN10
00		00		00		00	
01		01		01		01	
02		02		02		02	
03		03		03		03	
04	^	04		04		04	
05		05		05		05	
06	/	06		06		06	
07		07		07		07	
08	!	08		08		08	
09	!	09		09		09	
0A	.	0A		0A		0A	
0B		0B		0B		0B	
0C		0C		0C		0C	
0D		0D		0D		0D	
0E		0E		0E		0E	
0F		0F		0F		0F	
10		10		10		10	
11		11		11		11	
12		12		12		12	
13		13		13		13	
14	.	14		14		14	
15		15		15		15	
16		16		16		16	
17	/	17		17		17	
18		18		18		18	
19		19		19		19	
1A	.	1A		1A	α	1A	
1B	/	1B		1B	β	1B	
1C		1C		1C	γ	1C	
1D		1D		1D	δ	1D	
1E		1E		1E	ε	1E	
1F		1F		1F	ν	1F	
20		20		20		20	
21	/	21		21		21	
22	!	22		22		22	
23	/	23		23		23	
24	!	24		24		24	
25		25		25		25	
26		26		26		26	
27		27		27		27	
28	~	28		28		28	
29	~	29		29		29	

HEX	NNIO	HEX	TNIO	HEX	CNIO	HEX	SNIO
2A	✓	2A		2A		2A	
2B	°	2B		2B	p	2B	
2C	∪	2C		2C		2C	
2D		2D		2D		2D	
2E		2E		2E		2E	
2F		2F		2F		2F	
30	-	30		30		30	
31		31		31		31	
32		32		32		32	
33	•	33		33		33	
34	•	34		34		34	
35	•	35		35		35	
36	-	36		36		36	
37		37		37		37	
38	!	38		38		38	
39	!	39		39		39	
3A	-	3A		3A		3A	
3B	!	3B		3B		3B	
3C		3C		3C	m	3C	
3D		3D		3D	σ	3D	
3E	✓	3E		3E		3E	
3F		3F		3F		3F	
40		40		40		40	
41	2	41		41		41	
42	0	42		42		42	
43		43		43		43	
44	Δ	44		44		44	
45	Δ	45		45		45	
46	3	46		46		46	
47	!	47		47		47	
48	Γ	48		48		48	
49	□	49		49		49	
4A	[	4A	κ	4A		4A	κ
4B	•	4B	•	4B	•	4B	•
4C	•	4C	<	4C	<	4C	<
4D	(	4D	(	4D	(	4D	(
4E	•	4E	+	4E	+	4E	+
4F	!	4F	!	4F		4F	!
50	Δ	50	6	50		50	6
51	∫	51		51		51	
52	Σ	52		52		52	
53	•	53		53		53	
54	Λ	54		54		54	
55	~	55		55		55	
56	∇	56		56		56	
57	Σ	57		57		57	
58		58		58		58	
59	q	59		59		59	
5A	]	5A	!	5A		5A	!
5B	•	5B	\$	5B	\$	5B	\$
5C	•	5C	*	5C	*	5C	*
5D	)	5D	)	5D	)	5D	)
5E	!	5E	:	5E		5E	:
5F	~	5F	~	5F		5F	~
60	-	60	-	60	-	60	-
61	/	61	/	61	/	61	/
62	•	62		62		62	
63	Σ	63		63		63	
64	~	64		64		64	
65	Σ	65		65		65	
66	•	66		66		66	

Creating Micro Control Records via RHB195

The program RHB195 permits the user to create and write the required ID record on a tape without modifying his program. Because the ID record must be at least 56 bytes long, the LRECL of the Micro data set must be at least 56.

RHB195 must be run as a separate step before the step which writes data on the Micro tape. RHB195 creates and writes the ID record (from the information on the JOB statement) and a control record. The DCB parameters for the control record must match those used in the step which writes the data onto the tape. The receiving step which adds data to the tape must then specify a disposition of MOD, so the control records are not destroyed.

The message for the first control record is passed in the PARM= field of the step executing RHB195. This EXEC statement has the format:

```
//stepname EXEC PGM=RHB195,PARM='Control record without k&%& prefix'
```

Examples:

```
//stepname EXEC PGM=RHB195,PARM='M4500'
```

```
//stepname EXEC PGM=RHB195,PARM='MP360'
```

RHB195 supplies the necessary 'k&%&' in front of the control record. There can be no spaces within the PARM field between the initial quote and the control record information. If the number of bytes in the PARM field + 4 (for the k&%&) is greater than 56, this total becomes the minimum record length (LRECL) for the data set. If the user's data is shorter than this, he must pad each data record with blanks. A DD statement must be provided to describe the Micro tape. The ddname must be HEADER.

Additional control records may be specified through a SYSIN data set. The data set may be a DD * data set, or reside on tape or disk, but the LRECL should not be greater than 100. The control record specified by the PARM field will be written immediately after the ID record, followed by the control records specified by the records in the SYSIN data set. Either or both of these methods may be used at one time. If the user wishes to see a listing of the control records generated, he may include a //SYSPRINT DD SYSOUT=A statement.

In the following example, output from an ANS COBOL execution is directed to a Micro tape for output on microfiche with one extra original and five thermal duplicates.

Example:

```

//STEP1 EXEC PGM=RHB195,PARM='MP360'
//HEADER DD UNIT=9T1600,DISP=(NEW,PASS),VOL=PRIVATE,
// DSN=MICRO105.aaaaaaa.name,
// DCB=(RECFM=FBA,LRECL=133,BLKSIZE=3990)
//SYSIN DD *
1
D5
//STEP2 EXEC CBLACOMP
//COMP.SYSIN DD *
    (source program)
//STEP3 EXEC CBLALKGO,CORE=xxxK
//GO.OUTTAP DD DISP=(MOD,KEEP,DELETE),
// DSN=MICRO105.aaaaaaa.name

```

Step 1 executes the program RHB195. The mode control record in the PARM field asks for P360. The ddname must be HEADER, the disposition must be (NEW,PASS). DCB information must be supplied, and the LRECL subparameter of the DCB must be at least 56.

The SYSIN input supplies micro control statements requesting one additional original and five duplicates.

The third step executes a user program that will write data on the Micro tape. The disposition in this case is (MOD,KEEP,DELETE). The DCB given in the HEADER DD statement must agree with the DCB used by the program that will write on the tape, whether supplied by DD statement or coded in the program.

When initializing a multi-volume Micro data set which is to be written on scratch tapes with RHB195 in one step and passing it to another step, do not supply the volcount subparameter in the first step.

STEP 1

```

//HEADER DD UNIT=9T1600,DSN=MICRO105.aaaaaaa.STOR,
// DISP=(NEW,PASS),DCB=(RECFM=FBA,LRECL=133,
// BLKSIZE=3915),VOLUME=PRIVATE

```

STEP 2

```

//DD1 DD DSN=MICRO105.aaaaaaa.STOR,DISP=(MOD,KEEP),
// VOLUME=(PRIVATE,,1,10)

```

If the data set is to be larger than 5 volumes (there are 10 in the example), the volcount subparameter must be supplied in the step in which the data is written.

When using preassigned tapes, it is possible to add on to a multi-volume tape data set with DISP=MOD. A volume sequence number must be given in the step which MODs the tape. The example below illustrates how to use RHB195 to begin creation of a multi-volume data set and extend it in a later step.

To set up the data set:

```
//STEP1 EXEC PGM=RHB195,PARM='MP360'  
//HEADER DD DSN=MICRO105.aaaaaaa.name,UNIT=9T1600,DISP=(NEW,PASS),  
// VOL=(PRIVATE,SER=(000001,000002,000003)),  
// DCB=(RECFM=FBA,LRECL=133,BLKSIZE=2660)
```

To write the data:

```
//GO.OUTPUT DD DSN=MICRO105.aaaaaaa.name,  
// DISP=(MOD,KEEP,DELETE),VOL=(PRIVATE,,1)
```

Extreme caution must be exercised if the data set is to be MODed in more than one step. The user must be aware of which tapes are written on in each step. For instance, if in the first step that the tape is MODed, tape 1 is filled and tape 2 is partially filled, in the second step to MOD the data set the volume parameter must specify 2 in the volume sequence field, specifying that writing should resume on the second volume. This means that the user must have some idea of the quantity of output generated by each step. If the proper volume sequence number is not specified, results are unpredictable.

Creating the ID Record from a User's Program

There are three subroutines, RHB194 (PL/I), RHB197 (FORTRAN, COBOL), and SVC253 (Assembler), which may be invoked by a user's program to set up the ID record information in the correct form for Micro tapes. In calling each of the subroutines, the user must supply an 80 character (20 word) area into which the subroutine will place the complete formatted ID record including the k&% prefix. It is then the user's responsibility to write the contents of this area on the Micro tape and to create and write any other control records required. The following are the linkages to these subroutines in the various programming languages used at NIH. JOBID is a dummy name -- the user can choose any name.

FORTRAN :

DIMENSION JOBID (20)

.

.

CALL RHB197 (JOBID)

.

(Write out JOBID under a 20A4 format.)

COBOL :

WORKING-STORAGE SECTION.
01 JOBID PICTURE X(80).

.

PROCEDURE DIVISION.

CALL 'RHB197' USING JOBID.

.

PL/I :

DCL RHB194 EXTERNAL ENTRY;
DCL JOBID CHARACTER(80);

.

CALL RHB194(JOBID) ;

.

Assembler :

.

LA 1,ADCON
SVC 253

.

ADCON DC A(JOBID)
JOBID DS CL80

.

Again, it is the user's responsibility in all languages to write the ID record on the tape, and to create and write all other control records needed.

to another step, do not supply the volcount subparameter in the first step.

STEP 1

```
//HEADER DD UNIT=9T1600,DSN=PRINT.aaaaaaa.STOR,  
// DISP=(NEW,PASS),DCB=(RECFM=FBA,LRECL=133,  
// BLKSIZE=3915),VOLUME=PRIVATE
```

STEP 2

```
//DD1 DD DSN=PRINT.aaaaaaa.STOR,DISP=(MOD,KEEP),  
// VOLUME=(PRIVATE,,1,10)
```

If the data set is to be larger than 5 volumes (there are 10 in the example), the volcount subparameter must be supplied in the step in which the data is written.

When using preassigned tapes, it is possible to add on to a multi-volume tape data set with DISP=MOD; a volume sequence number of 1 must be given in the step where the data is first written. The example below illustrates how to use RHB195 to begin creation of a multi-volume data set and extend it in a later step. The example requests printing on a user-supplied 2-part form with a private carriage control loop.

To set up the data set:

```
//STEP1 EXEC PGM=RHB195,PARM='F USE iii123 2-PART AND iii001 LOOP'  
//HEADER DD DSN=PRINT.aaaaaaa.name,UNIT=9T1600,DISP=(NEW,PASS),  
// VOL=(PRIVATE,SER=(000001,000002,000003)),  
// DCB=(RECFM=FBA,LRECL=133,BLKSIZE=3990)
```

To write the data:

```
//GO.OUTPUT DD DSN=PRINT.aaaaaaa.name,  
// DISP=(MOD,KEEP,DELETE),VOL=(PRIVATE,,1)
```

Extreme caution must be exercised if the data set is to be MODed in more than one step. The user must be aware of which tapes are written on in each step. For instance, if in the first step that the tape is MODed, tape 1 is filled and tape 2 is partially filled, in the second step to MOD the data set the volume parameter must specify 2 in the volume sequence field, specifying that writing should resume on the second volume. This means that the user must have some idea of the quantity of output generated by each step. If the proper volume sequence number is not specified, results are unpredictable.

Creating the SPOUT ID Record from a User's Program

There are three subroutines, RHB194 (PL/I), RHB197 (FORTRAN, COBOL), and SVC253 (Assembler), which may be invoked by a user's program to set up the ID record information in the correct form for SPOUT tapes. In calling each of the subroutines, the user must supply an 80 character (20 word) area into which the subroutine will place the complete formatted ID record including the k&%& prefix. It is then the user's responsibility to write the contents of this area on the SPOUT tape and to create and write any other SPOUT control records required. The following are the linkages to these subroutines in the various programming languages used at NIH. JOBID is a dummy name -- the user can choose any name.

FORTRAN :

```

      DIMENSION JOBID (20)
      .
      .
      .

```

```

      CALL RHB197 (JOBID)
      .

```

```

      (Write out JOBID under a 20A4 format.)

```

COBOL :

```

      WORKING-STORAGE SECTION.
      01 JOBID PICTURE X(80).
      .

```

```

      PROCEDURE DIVISION.
      .

```

```

      CALL 'RHB197' USING JOBID.
      .

```

PL/I :

```

      DCL RHB194 EXTERNAL ENTRY;
      DCL  JOBID  CHARACTER(80);
      .

```

```

      CALL RHB194(JOBID) ;
      .

```

Assembler :

```

      .
      .
      LA      1,ADCON
      SVC     253
      .

```

```

      ADCON   DC      A(JOBID)
      JOBID   DS      CL80
      .

```

Again, it is the user's responsibility in all languages to write the ID record on the SPOUT tape, and to create and write all other control records needed.

Labels Available for Printing

All labels mentioned below are pre-gummed and are attached to sheets of waxed paper which contain the necessary pin-feed holes. The labels are arranged in from one to four rows across the sheets. Samples of the available labels may be seen at Output Distribution Services.

Labels must be requested in terms of their size. If the data set to be listed contains carriage control, a special carriage control loop must be supplied so the printer "knows" where the top of each label is. If the user does not use the standard control loop for a given label, he must go to Output Distribution Services and ask to have one made. Output Distribution Services will create the carriage control loop, label it with his initials, box number, and an identifier, and store it for him.

Types of Labels Available at the Central Facility

Width x Length in inches	Number Across	Print Lines per Label	No. of Charac. per Line	Beginning Column per Label**	Carriage Control Loop Needed	Control Character For Head of Form
2 3/4 x 7/16	1,4	2	26	2,31,60,89	716	1,2,3
2 3/4 x 15/16	1,4	5*	26	2,31,60,89	1516	1,2
2 3/4 x 1 7/16	1	8*	26	2	1716	1
3 1/2 x 7/16	1,3	2	34	2,38,74	716	1,2,3
3 1/2 x 15/16	1,3	5*	34	2,38,74	1516	1,2
3 1/2 x 1 7/16	1,3	8*	34	2,38,74	1716	1
4 x 7/16	1	2	38	2	716	1,2,3
4 x 15/16	1,3	5*	38	2,43,84	1516	1,2
4 x 1 7/16	1,3	8*	38	2,43,84	1716	1
5 x 7/16	1	2	49	2	716	1,2,3
5 x 15/16	1,2	5*	49	2,53	1516	1,2
5 x 1 7/16	1	8*	49	2	1716	1

*If the maximum number of lines or the maximum number minus one is to be printed, a request to begin on the first print line must be included in the forms message.

**Note that column one is occupied by carriage control.

As an alternative to using carriage control, blank lines may be left between labels. There is one line of waxed paper backing between each label; so the total number of lines per label is equal to the maximum number of print lines plus one. (WYLBUR users should specify NOPAGE in their LIST command when using this method.)

When using standard loop 716, 3 separate carriage control characters must be used to skip to head of form. The character for the first label (or row of labels) is 1, for the second is 2, the third is 3, the fourth is 1, etc. When using standard loop 1516 the characters 1 and 2 must be used alternately to skip to head of form, and for loop 1716 only 1's are used.

Employing one of the standard loops described above, the SPOUT control record to list a data set (perhaps an address list) on a single column of 3 1/2 by 1 7/16 inch labels is:

```
//stepname EXEC PGM=RHB195,  
// PARM='F USE 3 1/2 X 1 7/16-1 LABELS AND 1716 LOOP'
```

Using a standard loop to list a data set with as many as seven lines per label on 4-inch labels would require:

```
//stepname EXEC PGM=RHB195,  
// PARM='F USE 4 X 1 7/16-1 LABELS FIRST PRINT LINE'
```

For users of RHB195 who have had a private carriage control loop constructed for them by Output Distribution Services, the SPOUT control record to print on the above labels is:

```
//stepname EXEC PGM=RHB195,  
// PARM='F USE 4 X 1 7/16-1 LABELS AND LOOP ident'
```

where "ident" is the identifier assigned to the loop by Output Distribution Services.

11.10.5 User-Supplied Forms and Cards

The user may bring in his own special paper, give it to Output Distribution Services, and then request it via a SPOUT message. The boxes will be labelled with the form number and the user's name and box number. It is the responsibility of the user to make sure that Output Distribution Services has enough of the special forms to print his SPOUT tape. If the supply of user's special forms is exhausted in the middle of a SPOUT run, the run will be cancelled and the user will be charged for the run up to that point, and the tape will be saved until the user can be contacted.

The Computer Center's online punching service supplies only plain manila cards. SPOUT must be used if user-supplied cards (e.g. preprinted or colored) are needed.

User-supplied cards are handled in the same way as user-supplied forms.

11.10.6 SPOUT Control Records for Printer/Punch

Following is a summary of the SPOUT control records for the printer/punch:

Forms	F	operator message
Punch	P	operator message
Train	Txx10	operator message
Lines-per-Page	Lnnn	message ignored
Burst and Decollate	B	output handling message
Copies	1-3 digits	operator message
Printer Spacing	Snn	operator message
Photo Quality	Q	no message permitted

Particularly for users of RHB195, it is good to avoid using an unnecessary number of control records. Note that multiple copies, special paper, and a special carriage control tape may all be requested with one control record (COPIES). Similarly, special train and special forms may be requested with one control record (TRAIN).

Examples using RHB195:

```
//S2 EXEC PGM=RHB195,PARM='10 USE iiiRPT PAPER AND iii3 LOOP'
      (for 10 copies using user forms and a private
      carriage control loop)
```

```
//S04 EXEC PGM=RHB195,PARM='TMN10 USE 900 PAPER'
      (for the MN10 print train and special paper)
```

The following are examples of instances where more than 1 control record will be needed in a 'group' in order to fulfill the requirements for printing the tape:

- A. A special train and multiple copies are needed. This requires a TRAIN and a COPIES record.
- B. Any time a LINES or BURST record is used, some other control record must appear immediately before or after it.
- C. If a BURST record is used to request decollating, a FORMS record is also required to specify the user-supplied multi-part form.

12.1 PUBLICATIONS

The following manuals are referenced in the Utility Section. These manuals are available through the Technical Information Office:

A. IBM OS/VS Utilities, Form No. GC26-3902

This publication discusses the capabilities of the IBM System/370 Operating System utility programs and the control statements used with each program.

B. IBM OS/VS Message Library: Utilities Messages, Form No. GC26-3920

This publication contains explanations of messages produced by IBM utilities.

C. IBM OS/VS2 MVS JCL, Form No. GC28-0692

This publication describes the facilities provided with the Job Control Language and contains the information necessary to code Job Control Language statements.

D. IBM OS/VS Sort/Merge Programmer's Guide, Form No. SC33-4035

This publication contains specifications for the IBM System/370 Operating System sort/merge programs, including control statement preparation, program operation, and I/O device assignment.

E. IBM OS/VS Linkage Editor and Loader, Form No. GC26-3813

This publication provides the information necessary to use the Linkage Editor or Loader program to prepare the output of a language translator for execution.

F. IBM OS/VS Message Library: Linkage Editor and Load Messages, Form No. GC38-1007.

This publication contains explanations of messages produced by the Linkage Editor and Loader.

G. IBM OS/VS Data Management Services Guide, Form No. GC26-3875

This publication describes all IBM data management except for VSAM, TSO, OCR and other specialized applications.

H. WYLBUR documentation

These publications describe the facilities available in WYLBUR, a computer program which functions as a text editor, command processor, document formatter and remote job entry facility. A list of the available WYLBUR publications is given in Section 3.2.

I. Edit Format Utility Package

This manual describes a package of programs for reading and writing edit format data sets from TSO programs and batch jobs. WYLBUR creates data sets in edit format.

J. PALTape--A Utility for Handling Tape Problems

This publication provides the information necessary to use the PALTape procedure to copy, print or analyze a problem tape.

K. Easy Reader - A Private Procedure Library Facility

This publication describes how a user can create a private library of JCL cataloged procedures.

Example 21

```
// (class B JOB statement)
// *MESSAGE serial,RS;serial,W
// *KEYWORD=kkk
// stepname EXEC DSCOPY,NAME='dsname',INTAPE=serial,INLBL=NL,
//   INSTOR=9T1600,NEWNAME='aaaaiii.dsname',OUTTAPE=serial,
//   OUTSTOR=9T1600,RECFM=FB,LRECL=80,BLKSIZE=800
```

12.4.3 Printing Data Sets

The following chart summarizes the print utilities available at the Computer Center. Consult the additional documentation cited in the chart before attempting to use a utility for the first time.

Summary of Print Utilities

Procedure Name	Documentation	Input	Function
PRINT (IBM utility IEBPTPCH)	12.4.3 <u>IBM OS/VS Utilities</u> GC26-3902	Sequential or Partitioned data set	Prints data set; can select and format material printed.
DSLIS	12.4.3	Sequential data set	Prints any non-EDIT format data set from tape or disk.
USRPDSL	12.6.7 12.13.2	PDSs or user's members from SYS1.PROCLIB	Prints contents of PDS members or member names only, with space and directory blocks allocated and used. No editing.
EDSLIS	12.4.3	EDIT format sequential data set	Prints entire data set; no editing.
PDSLIS	12.6.9	Partitioned data set	Prints entire data set; no editing.
ISLIS	12.6.9	Indexed sequential data set	Prints entire data set; no editing.
DDSLIS	12.6.9	Direct data set	Prints entire data set; no editing.
PALTAPE	<u>PALTAPE Manual</u> 12.5.3	Tapes with problems	Not a production tool; to be used for problem tapes only.

PRINT

Under the standard format, each new logical record begins on a new line. The end of a logical record is marked by an asterisk and the end of a block by two asterisks. Each line contains up to 96 characters, divided into 12 groups of 8 characters each, separated by two blanks. Therefore, if the input to the basic print operation is:

THIS IS WHAT THE DEFAULT OUTPUT LOOKS LIKE

the output will be:

THIS IS WHAT THE DEFAULT OUTPUT LOOKS LIKE*

Unprintable characters appear as blanks. The data can be printed in character or hexadecimal format or converted from packed decimal.

The default DCB attributes of the SYSUT2 output data set are RECFM=FBA,LRECL=121,BLKSIZE=121. Note, the sum of the field lengths in any IEBTPCH record statement may not exceed the output logical record length minus 1. Also, the maximum linesize is 144.

The DD statement SYSUT1 defines the input data set. The procedure assumes sequential input and no editing on the output data set. The stepname is PRINT.

For more information on the IEBTPCH utility control statements, see the IBM OS/VS Utilities manual, Form No. GC26-3902.

Example 22:

```
// (JOB statement)
/*MESSAGE serial,RS
/*KEYWORD=kkk
/*ROUTE XEQ 7TRACK
//stepname EXEC PRINT
//PRINT.SYSUT1 DD UNIT=7TRACK,DISP=(OLD,KEEP),
//      DSN=aaaaaaa.name,VOL=(PRIVATE,SER=serial),
//      LABEL=(1,NL),
//      DCB=(RECFM=FB,LRECL=80,BLKSIZE=3120,DEN=1,TRTCH=ET)
```

In this example the first data set on a 7-track unlabeled special tape will be dumped to the printer. Since there is no label, DCB information must be supplied. If a standard-labeled tape is used, the DCB information is not needed on the DD statement. The data will be printed under standard specifications.

Frequently users want to print a tape, editing the data to eliminate the blanks between each group of eight characters. The

following example illustrates how to do this and start printing after the 250th record. In this example the FIELD parameter takes advantage of all the defaults for starting input and output locations. It will read the first 80 bytes of the input record and start writing in the first position on the print line.

Example 23:

```
//stepname EXEC PRINT
//PRINT.SYSUT1 DD DSN=aaaaaaa.name,UNIT=9T1600,
//  DISP=(OLD,KEEP),VOL=(PRIVATE,SER=serial)
//PRINT.SYSIN DD *
PRINT STRTAFT=250,MAXFLDS=1
RECORD FIELD=(80)
```

No editing is allowed when printing variable spanned (VS) or variable blocked spanned (VBS) records. These data sets must be printed with the 8 character, 2 blank format.

If the LRECL of the data set is greater than 144, editing can still be done if the user chooses as the LRECL a valid linesize (no more than 144) which can be evenly divided into the blocksize. In the case of a data set with an LRECL of 260 and a blocksize of 5200, each logical record can be broken into 2 print lines of 130 characters each. To accomplish this, DCB parameters must be specified for the input (SYSUT1) data set and IEBPTPCH control statements must be provided. DCB parameters must also be provided for the output (SYSUT2) data set since they exceed the default (121 output record characters per line). Notice that the LRECL is increased by 1 to allow for the carriage control.

Example 24:

```
//stepname EXEC PRINT
//PRINT.SYSUT1 DD DSN=aaaaaaa.name,DISP=SHR,
//  UNIT=FILE,VOL=SER=volser,
//  DCB=(RECFM=FB,LRECL=130,BLKSIZE=5200)
//PRINT.SYSUT2 DD DCB=(RECFM=FBA,LRECL=131,BLKSIZE=131)
//PRINT.SYSIN DD *
PRINT MAXFLDS=1
RECORD FIELD=(130)
```

This method may be used for data sets with either fixed or fixed blocked records. If the data set has fixed-length unblocked records, RECFM=FB must be specified on the SYSUT1 DD statement.

"optional."

Please note that in addition to the utilities listed above, the following, which are described elsewhere in Section 12, can also be used with tape data sets:

Procedure	Documentation
COPY	12.4.2
PRINT	12.4.3
PUNCH	12.3.2
COMPARE	12.4.5
BIGSRT	12.4.4
LITSRT	12.4.4
MERGE	12.4.4

The following table may aid the user in deciding which utility to choose depending upon the function to be performed.

Where a choice of utilities is given, USE THE FIRST ONE LISTED unless a special feature of the other is required.

FUNCTION

UTILITY

1. Clean or degauss a tape
 - A. Special tapes (in 12A, B-61)
 - B. NIH SL or U tapes (see Info. Media Librarian)
2. Compare two data sets COMPARE
3. Copy tape to tape
 - A. SL to SL
 - (1) all files, single reel only TAPECOPY
 - (2) selected file(s) DSCOPY, COPY, PALTape
 - B. SL to NL DSCOPY, COPY, PALTape
 - C. NL to SL DSCOPY, COPY, PALTape
 - D. NL to NL
 - (1) all files, single or multireel TAPECOPY
 - (2) selected file(s) DSCOPY, COPY, PALTape
 - E. AL to SL or NL DSCOPY, COPY, PALTape
- F. Notes on copying tapes:

Only one 7-track, one 9-track 6250 BPI, and one 9-track 800 BPI tape may be mounted at a time. Each requires a /*ROUTE XEQ statement.

ASCII data may be written only on a user-supplied (special) 9-track tape.

TAPECOPY - all data sets are copied in one step.

COPY - requires one step per data set to be copied.

PALTape - requires control statements for each data set

to be copied.

- G. Special features for copying -
 - (1) Skip to end of file PALTape
 - (2) Backspace a number of blocks PALTape
 - (3) Begin at any physical block or logical record PALTape
 - (4) Reblock COPY, PALTape
 - (5) Tape with I/O errors
 - a. selected data set(s) PALTape (may specify error option)
 - b. all data sets TAPECOPY (reduced error recovery and/or bypass bad blocks)
 - (6) Check output tape TAPECOPY (two possible methods)
- 4. Determine density for 9-track TAPeMAP
- 5. Find out if a tape is labeled TAPeMAP
- 6. Determine TRTCH and DEN for 7-track PALTape
- 7. Print
 - A. Tape labels TAPeMAP, PALTape
 - B. Data set labels TAPeMAP, PALTape
 - C. Specified file in hexadecimal PRINT, PALTape
 - D. Selected records PRINT, PALTape
 - E. Specified files DSLIST, PRINT, PALTape
 - F. Begin at any record in any block PALTape
 - G. Tape with I/O errors PALTape
 - H. Number of blocks and largest block size for each file TAPeMAP
- 8. Process SL input tape as if NL PALTape
- 9. Punch
 - A. Specified file PUNCH
 - B. Selected records PUNCH

10. Put tape mark on new NL tape
 - A. Special tapes (see Output Distribution Services)
 - B. NIH tapes (see Info. Media Librarian)
11. Re-label SL tape with new VOLSER
 - A. Special tapes (see Output Distribution Services)
 - B. NIH SL tapes (see Info. Media Librarian)
12. Change status of a tape
READ or WRITE, PUBLIC or
NOPUBLIC TAPESTAT
13. Prevent release of an
inactive tape TAPEUSE
14. Release a tape (see Tape/Disk Report)
15. Sort/Merge LITSRT, BIGSRT, MERGE
16. Update, replace or insert
records IEBUPDTE (tape to tape)

TAPEMAP, TAPECOPY, TAPESTAT, TAPEUSE, and PALTAPE are NIH-written routines. COPY is the cataloged procedure for IEBGENER; PRINT and PUNCH use IEBPTPCH; COMPARE uses IEBCOMPR. TAPEMAP, TAPECOPY, and PALTAPE should all be used in a single-step job and should not be mixed with steps using standard data management routines.

PALTAPE is to be used only for tapes that cannot be processed by standard utilities. It is not intended to be a production tool.

Other utilities available for saving/retrieving disk/card data sets on tape are described elsewhere in this section. See the IBM OS/VS Utilities manual, Form No. GC26-3902, for IEBUPDTE.

12.5.2 Dumping the Labels of a Tape

TAPEMAP is used to format the volume, header, and trailer labels of a standard-labeled tape for one or more data sets on a tape. Appropriate messages are printed for error conditions such as I/O errors, data checks, incorrect density, tape marks embedded in data, and missing labels. The program executes against logical data sets. That is, it treats the header labels, actual data, and trailer labels as a single entity. Up to 256 files (data sets) may be mapped with just one execution of the TAPEMAP procedure. The first page of TAPEMAP output includes a brief explanation of how labels are formatted, and a map of the volume (VOL1) label of the tape (if the tape is standard-labeled).

Each label is printed in two formats. The first describes the different fields of the labels. The second format consists of three print lines of 80 characters each. Line 1 contains the character representation of the label as it exists on tape. Lines 2 and 3 are the hexadecimal equivalent of the character directly above on line 1. This hexadecimal equivalent is printed vertically for ease of association. The first logical data record of each data set requested is printed up to a maximum of 132 characters. It is formatted in three output lines: character data in line 1, and the equivalent hexadecimal representation in lines 2 and 3.

TAPEMAP will handle OS, DOS, and ASCII standard-labeled tapes and unlabeled tapes, but will reject non-standard labels.

If SCAN is specified, the physical block count and the size of the largest block will be printed along with the character representation and hexadecimal equivalent of the first logical data record of each requested data set. For unlabeled ASCII tapes, the character representation will be incorrectly printed.

TAPEMAP can handle only one tape volume at a time. It is not designed to be used as a step within a job that makes other references to the tape. It should be used as a single-step diagnostic tool only.

The symbolic parameters available to TAPEMAP are:

TAPE - Required

Supplies the volume serial number of the tape from which the data is being mapped.

DRIVE - Optional

Supplies the unit type of the tape device being used. Must be coded if other than 1600 BPI tape is being used. Defaults to DRIVE=9T1600.

12.6.3 Recovering Lost Data Sets

The ADSRECOV procedure (for Any Data Set RECOVer) restores a data set from a system backup tape (of a FILE or PDS pack) or from a user's backup of a private pack. The procedure can also restore all data sets from a user's backup tape and either add them to the existing contents of the private disk pack or replace the contents of the pack with the contents of the backup tape. Please note, however, that DISKGET (Section 12.7.4) is more efficient than ADSRECOV to replace the contents of a disk with those of a 3330 backup tape.

WYLBUR's RECOVER command, which uses the ADSRECOV procedure, provides a convenient way to recover any data set which can be used with WYLBUR (e.g. not BDAM and ISAM).

Each time a disk pack is dumped (e.g. with DISKSAVE), a table containing the dates and times of each backup is updated. (Currently, the FILE and PDS packs are backed up twice weekly, before 8:00 a.m., Wednesday and Saturday. These tapes are kept for approximately three weeks before being recycled.)

The ADSRECOV job submitted to the system by the user does not perform the actual recovery itself. Instead, this job will create and submit to the system another job which will actually perform the recovery process for the data set. Thus, the user submitting an ADSRECOV job, which includes at least one RECOVER or RECOVER ALL control statement, will receive two pieces of printed output:

- 1) The output of the ADSRECOV job, which contains information concerning the job that it submitted (such as the job number and exactly which data sets will be recovered by the job).
- 2) The output of the job that performed the actual recovery, which will tell the user if each data set was recovered properly and, if not, the problems encountered.

The class of the recovery job submitted by ADSRECOV depends on the mountable devices it requires as follows:

Devices required	Class for recovery job
no mountable disks, one or two tapes	Class F
one mountable disk, one tape	Class B
one or more mountable disks, more than one tape	Class C

The following charts outline the functions of the RECOVER, RECOVER ALL, LIST, and SET control statements.

RECOVER Functions	RECOVER Options	R/O*	Data Set Organization
Causes ADSRECOV to submit a job which restores a data set to disk from a backup tape.	NAME	R	Sequential
	DISK	R	Partitioned
	NEWDISK	O	Direct Access
	NEWNAME	O	Indexed Sequential
	DATE	O	
	STATUS	O	
	TAPE SPECIAL DRIVE	R O O	(Additional only if volume serial number of a particular tape is specified)
	UNITS SIZE INCR	O O O	Additional for Sequential
	DSORG=PO DIR UNITS SIZE INCR UNUSED	R O O O O O	Additional for Partitioned
	DSORG=DA	R	Additional for Direct Access
	DSORG=IS SIZE INDEX OVFLOW	R O O O	Additional for Indexed Sequential

*R = Required; O = Optional

Retrieving Multiple Data Sets with a Single Job

Multiple data sets may be retrieved in one job by placing /*RETRIEVE END statements between the /*RETRIEVE statements for each data set. For example, if the data sets 22/1, 22/32 and aaaa111.MYDATA are to be retrieved, the following job would be submitted:

Example 96:

```
// (class F JOB statement)
/*KEYWORD=kkk
/*RETRIEVE 22/1
/*RETRIEVE END
/*RETRIEVE 22/32
/*RETRIEVE END
/*RETRIEVE aaaa111.MYDATA
```

In general, about 10 data sets may be retrieved per job using the above method. If more are to be retrieved, additional class C jobs should be submitted.

WYLBUR's RETRIEVE command also makes provision for retrieving multiple data sets with a single job.

12.6.5 Renaming Disk Data Sets

The procedure DSRENAME enables users to rename data sets on the FILE or PDS disks as well as data sets stored on private user disk packs. This procedure may be used to rename sequential, partitioned, direct and indexed-sequential data sets. The TSO and WYLBUR RENAME commands may also be used to perform this function.

The symbolic parameters used in DSRENAME are:

NAME= is used to specify the current dsname, i.e.: the name of the data set to be renamed. Required.

NEWNAME= is used to specify the new data set name. Required.

DISK= is used to specify the volume serial number of the disk pack on which the data set resides (e.g.: FILE23 or D00555). Required.

STORAGE=DISK is used only to indicate that the data set is on a private, mountable pack. Coded exactly as shown. Default is STORAGE=FILE; this supplies a value for UNIT. Optional.

Example 97: To rename a data set stored on FILE25.

```
//stepname EXEC DSRENAME,NAME='aaaaiii.oldname',
// NEWNAME='aaaaiii.newname',DISK=FILE25
```

Example 98: To rename a data set stored on user's private mountable disk pack D00001.

```
// (JOB statement)
/*MESSAGE D00001
/*KEYWORD=kkk
//stepname EXEC DSRENAME,NAME='aaaaiii.oldname',
// NEWNAME='aaaaiii.newname',STORAGE=DISK,DISK=D00001
```

A step completion code of 000 indicates that the rename operation has completed successfully. Error conditions are indicated by a step completion code of 008 and an explanatory message in the printed output of the DSRENAME step.

2/27/81

PAGE 577

"R" or "O" in the right-hand column in the above chart indicates whether the symbolic parameter is "required" or "optional."

12.8.2 Reading and Writing EDIT Format Data Sets

Procedures have been developed which allow the WYLBUR user to manipulate data sets stored in EDIT format, the normal form for a WYLBUR data set. Programmers should use these procedures and not try to write their own to convert to and from EDIT format. The Computer Center will not pre-announce internal changes to the EDIT format software, and assumes no responsibility if changes to that software cause a job that invokes that software illegally to fail. As an example, the internal format of WYLBUR's EDIT format data sets is subject to change without advance notice.

Procedures to sort (EDSSORT), list (EDSLIST), punch (EDSPUNCH), put cards on disks (EDSCARDS), and store and retrieve data sets on tape in EDIT format are described in other areas of the Utility Section. The following procedures are also very useful to the WYLBUR user.

Accessing EDIT Format Data Sets from Batch Jobs (EDSIN, EDSOUT)

A. EDSIN - Reading EDIT Format Data Sets

EDSIN is used to read an EDIT format data set. It converts the data set to fixed-blocked format and stores it into a temporary data set named &INPUT which is PASSED to a later job step.

The procedure uses the following symbolic parameters:

NAME='aaaaiii.dsname' - Required

Specifies the name of the EDIT format data set which will be input to the program.

DISK=fileser - Required

Specifies the volume serial number of the disk where the EDIT format data set is stored.

LRECL=record - Optional

Specifies the logical record length for the temporary data set. Cannot exceed 500. LRECL=80 is the default.

BLKSIZE=block - Optional

Specifies the blocksize for the temporary data set. It must be a multiple of the LRECL. BLKSIZE=3120 is the default.

STORAGE=DISK - Optional

Specifies that the EDIT format data set is stored on a private mountable disk. STORAGE=FILE is the default; this supplies a value for UNIT.

CC=carriage control - Optional

Specifies that the temporary data set is to have carriage control. CC=A is used for ASA carriage control; CC=M is used

for Machine carriage control. No carriage control is the default.

INPUT='&temp' - Optional

Specifies a name for the temporary input data set.
INPUT='&INPUT' is the default.

OPTIONS='options list' - Optional

The options which may be specified are: LIST to create a listing of the input data set and NUMBERED to place line numbers in columns 73-80 if the data set is being converted to LRECL=80.

For example, the following Job Control Language statements can be used to compile a program which is stored as a WYLBUR data set:

Example 123:

```
//stepname EXEC EDSIN,NAME='aaaaa.ii.dsname',  
//          DISK=filesr  
//stepname EXEC 11lvCOMP  
//COMP.SYSIN DD DSN=&INPUT,DISP=(OLD,DELETE)
```

where "111" is the language prefix
 "v" is the version
 (e.g. FORGCOMP for FORTRAN G1)

The following Job Control Language permits a user's program to read data from an EDIT format data set:

Example 124:

```
//stepname EXEC EDSIN,NAME='aaaaa.ii.dsname',  
//          DISK=filesr  
//stepname EXEC 11lvLKGO  
//GO.SYSIN DD DSN=&INPUT,DISP=(OLD,DELETE)
```

where "11lvLKGO" is the name of the procedure for the
 link-edit and run step (ASMFLKGO,
 CBLALKGO, etc.)

EDSIN normally changes the data set to card images. To change it to records of some other length, use:

Example 125:

```
//stepname EXEC EDSIN,NAME='aaaaa.ii.dsname',  
//          DISK=filesr,LRECL=record,BLKSIZE=block
```

where "record" is the logical record length (may not be

greater than 500)
"block" is the block size. It must be a multiple
of the "record".

Recommended block sizes for commonly used "record" (LRECL)
values are:

LRECL	BLKSIZE
80	6160
120	6120
133	6118

When converting an EDIT format data set to card images, the
line numbers can be placed in columns 73 through 80:

Example 126:

```
//stepname EXEC EDSIN,NAME='aaaaiii.dsname',  
//          DISK=fileser,OPTIONS=NUMBERED
```

A listing of the data set can also be obtained.

Example 127:

```
//stepname EXEC EDSIN,NAME='aaaaiii.dsname',  
//          DISK=fileser,OPTIONS=LIST
```

The LIST and NUMBERED options can be combined.

Example 128:

```
//stepname EXEC EDSIN,NAME='aaaaiii.dsname',  
//          DISK=fileser,OPTIONS='LIST,NUMBERED'
```

A temporary name other than &INPUT can be used.

Example 129:

```
//stepname EXEC EDSIN,NAME='aaaaiii.dsname',  
//          DISK=fileser,INPUT='&temp'
```

where "&temp" is the temporary name to be used. Remember
that it must be 8 or fewer characters and
must start with an ampersand.

This is useful if a job step is to read several EDIT format data
sets since each data set must have a different dsname.

B. EDSQUI - Writing EDIT Format Data Sets

EDSQUT is used to convert output from a batch job to EDIT format. It assumes that the output is in a temporary data set named &OUTPUT which is PASSED from an earlier job step. Because the line numbers of the EDIT format data set are integers, no lines after 9,999 can be accessed by WYLBUR. The logical record length of a data set which is input to EDSQUT should not exceed 500 (504 for Variable format) and the blocksize should not exceed 13,030.

The procedure uses the following symbolic parameters:

NAME='aaaaiii.dsname' - Required

Specifies the name of the EDIT format data set which will be stored on disk.

DISK=fileser - Required

Specifies the volume serial number of the disk where the EDIT format data set is to be stored.

STORAGE=DISK - Optional

Specified only if the disk is a private mountable pack. STORAGE=FILE is the default; this supplies a value for UNIT.

TYPE=OLD - Optional

Indicates that a data set with the same name is to be replaced. TYPE=NEW is the default.

SIZE=amount - Optional

Specifies a maximum number of lines for the temporary data set. SIZE=1000 is the default.

OUTPUT='&temp' - Optional

Specifies a name for the temporary output data set. OUTPUT='&OUTPUT' is the default.

DISP=PASS - Optional

Specifies that the temporary data set is to be used in subsequent steps. DISP=DELETE is the default.

For example, to convert normal printer output from a FORTRAN-G1 Program to EDIT format, use the following Job Control Language:

Example 130:

```
// (JOB statement)
/*KEYWORD=kkk
//stepname EXEC FORGCOMP
//COMP.SYSIN DD *
(FORTRAN-G1 program)
/*
//stepname EXEC FORGLKGO
//GO.FT10F001 DD DSN=&OUTPUT,UNIT=SYSDA,
// SPACE=(TRK,(30,10)),DISP=(,PASS),
// DCB=(RECFM=FB,LRECL=133,BLKSIZE=3990)
//stepname EXEC EDSOUT,NAME='aaaaiii.dsname',
// DISK=fileser
```

A data set which already exists can be overwritten:

Example 131:

```
//stepname EXEC EDSOUT,NAME='aaaaiii.dsname',
// DISK=fileser,TYPE=OLD
```

The default space allocation for EDSOUT can be changed:

Example 132:

```
//stepname EXEC EDSOUT,NAME='aaaaiii.dsname',
// DISK=fileser,SIZE=amount
```

where "amount" is the expected number of lines of output.

A different temporary data set name can be specified:

Example 133:

```
//stepname EXEC EDSOUT,NAME='aaaaiii.dsname',
// DISK=fileser,OUTPUT='&temp'
```

where "&temp" is the temporary name to be used. Remember that it must be 8 or fewer characters and must start with an ampersand.

This is useful if a job step has several output data sets which are to be converted to EDIT format since each one must have a different dsname.

The default step-end disposition is DELETE for the input data set. If the data set is to be used in subsequent steps, DISP=PASS should be coded.

Example 134:

```
// EXEC EDSOUT,NAME='aaaaiii.dsname',DISK=fileser,
// DISP=PASS
```


Storing GO Step Output into a WYLBUR Data Set (EDSOUT)

The following general example of Job Control Language statements can be used to store GO step output into a WYLBUR data set:

Example 135:

```
//stepname EXEC 111vLKGO
//GO.ddname DD UNIT=SYSDA,DISP=(NEW,PASS),
// DCB=(RECFM=format,LRECL=record,BLKSIZE=block),
// SPACE=(TRK,(10,5)),DSN=&OUTPUT
//GO.SYSIN DD *
      (data)
//stepname EXEC EDSOUT,NAME='aaaaiii.dsname',
//          DISK=filesr
```

where "111vLKGO" is the name of the link-edit and go procedure for the language being used (e.g. FORGLKGO, CBLALKGO)

"ddname" is the name for the printer either defined by the program or defined by the language

"format" is the format of the created data set, including the kind of carriage control which is either defined in the program or defined by the language

"record" is the number of characters per line of output plus 1 for a carriage control character, and can be changed by the programmer; it must be less than or equal to 500

"block" is an integral multiple of "record." The largest value which does not exceed 6233 is recommended.

Specific information about the "standard" output formats may be found in the appropriate language section.

12.9 USING THE LINKAGE EDITOR

The Linkage Editor is a service program used to prepare loadable programs from object modules (the output of language translators such as FORTRAN, COBOL, or Assembly Language). Link-editing is a necessary step that follows source program assembly or compilation.

The basic function of the Linkage Editor is to link separately assembled or compiled object modules into one load module. The load module is in a format suitable for loading and execution. The primary input to the Linkage Editor consists of object or load modules and Linkage Editor control statements. The Linkage Editor may also construct load modules from data sets other than its primary input. The programmer, through the use of Linkage Editor control statements, may specify a library (or libraries) to be searched in addition to those that are searched automatically. Linkage Editor control statements also may aid program modification by replacing, deleting, and rearranging control sections. This means that an entire program need not be recompiled when there are changes to only a few of its control sections.

The Linkage Editor can also be used to construct overlay programs. This permits the programmer to decrease the amount of core a program requires by allowing those modules immediately needed to overlay modules not needed at that time.

The Linkage Editor also reserves storage for the common control sections generated by the Assembler and FORTRAN languages and the static external areas that are generated by PL/I.

Finally, processing options and a log of diagnostic error messages are provided to aid the programmer. For additional information on the Linkage Editor and its functions, consult the manual, IBM OS/VS Linkage Editor and Loader, GC26-3813.

The Linkage Editor issues a RESERVE to the disk pack which will contain the resultant load module. This RESERVE is a hardware command that prevents any other subsystem from accessing that entire disk pack. This is done to insure that multiple jobs running on different subsystems do not attempt to update the same data set concurrently. Under normal circumstances this lockout lasts a very short time and has little detrimental effect on the other machines. However, very large link edits can cause severe lockout situations. These large load modules should be placed on the PDS packs and not on the FILE packs. Any link edit where the lock out time exceeds several minutes should have the output load module placed on the PDS packs. The beginning and end lock out time period are indicated by the ***RESERVE and ***RELEASE message in the job's JES2 job log.

sequence number must be given after the serial number, i.e., VOL=device=(serial,seqno).

The procedure, UNCATDS, is used to uncatalog data sets. The control statement has the format:

Col.2

```
UNCATLG DSNAME=NIHD.aaaaiii.name
```

DSNAME must be spelled as shown and must specify the fully qualified name of the data set. Volume information is not required when uncataloging a data set.

Example 145:

```
//stepname EXEC UNCATDS
//SYSIN DD *
UNCATLG DSNAME=NIHD.aaaaiii.name1
UNCATLG DSNAME=NIHD.aaaaiii.name2
UNCATLG DSNAME=NIHD.aaaaiii.name3
UNCATLG DSNAME=NIHD.aaaaiii.name4
```

This procedure does not delete or scratch a data set; it merely removes the entry from the system catalog. If the data set is retrieved by the catalog and given a disposition of DELETE, it will also be uncataloged.

The procedure CATLIST is used to list the names of cataloged data sets belonging to a particular user. In its simplest form the names of the cataloged data sets belonging to the user whose initials appear in the jobname field of the JOB statement will be printed.

Example 146:

```
//iiiname JOB (aaaa,box,class),user
/*KEYWORD=kkk
//stepname EXEC CATLIST
```

The names of the cataloged data sets belonging to any account/initials combination may be listed if the keywords are specified. They are requested on control statements supplied as SYSIN input.

Example 147:

```
//stepname EXEC CATLIST
//SYSIN DD *
AAAAIII/KKK
BBBBJJJ/LLL
/*
```

In addition to the data set names, CATLIST prints and interprets the device type code in each data set's catalog entry.

TSO and WYLBUR data sets are the only exceptions to the above naming restrictions. They are cataloged in a separate catalog and the format of their names is "aaaaiii.dsname" where aaaa is the account number and iii are the registered initials. The NIHD prefix is not used.

To be used by TSO, a data set must be cataloged. Data sets created through TSO are automatically cataloged, but data sets created via WYLBUR or the batch must be cataloged explicitly. WYLBUR, TSO, or the TSOCAT procedure illustrated below may be used.

Example 148:

```
//          (class A JOB statement)
/*KEYWORD=kkk
//stepname EXEC TSOCAT,NAME='aaaaiii.dsname',DISK=fileser
```

To uncatalog the data set, use the following:

Example 149:

```
//stepname EXEC TSOCAT,NAME='aaaaiii.dsname',TYPE=UNCAT
```

The separate catalog for TSO and WYLBUR data sets is only accessible by WYLBUR, TSO and the TSOCAT procedure. Thus, batch jobs which are accessing WYLBUR or TSO data sets must give complete volume and unit information.